Trabajo de Fin de Grado

Aprendizaje automático de modelos de atención visual como apoyo a la videovigilancia en el transporte público

Sergio Parejo López

Grado en Ingeniería de sistemas Audiovisuales

Tutor:

Iván González Díaz

A mis padres

Índice

Abstract

Los sistemas que modelan la atención visual humana están llevando a cabo su desarrollo gracias a la tecnología que permite la adquisición de características basadas en tareas, dando lugar a los modelos conocidos como *top-down.*

En nuestro proyecto, vamos a intentar crear uno de ellos basado en el ámbito de la videovigilancia del transporte público, tratando de conseguir analizar las zonas de las imágenes que atraen nuestra atención visual cuando nos enfrentamos a dicha tarea. Para ello, haremos uso de un conjunto de árboles de clasificación ya que necesitamos unas prestaciones elevadas que permitan acercarse a implementaciones en tiempo real.

Además, será parte principal de este proyecto la creación de una base de datos que nos permita desarrollar un sistema visual lo más óptimo posible ya que, para el objetivo propuesto no existen bases de datos abiertas. Este estudio analítico trata de inferir un modelo que, a partir de la muestra utilizada, se pueda aplicar en la mejor medida posible a sistemas de videovigilancia que tomen parte en el transporte público. Por eso debemos intentar generalizar el conjunto lo máximo posible, haciendo uso, para ello, de clasificación supervisada tipo *Random Forest*

Por último, compararemos nuestros resultados con otros modelos presentes para establecer conclusiones sobre el modelo utilizado y los puntos a mejorar.

1. Introducción

Con la evolución del transporte público, los sistemas de seguridad que están integrados en los mismos han sufrido una evolución paralela, mejorando y buscando las últimas posibilidades tecnológicas del mercado. Este hecho, directamente, ha repercutido en la introducción de circuitos de videovigilancia prácticamente en cualquier medio de transporte público del planeta, ya que, es también función de estos garantizar la seguridad y la experiencia de viaje de sus usuarios.

Esto nos sitúa en un escenario con inmensas capacidades de cara al procesamiento de la imagen y el video, ya que constantemente se están capturando imágenes que dejan de ser irrelevantes en el momento que algún pasajero sufre, de manera fortuita o provocada, una situación anómala. En este proyecto, nos enfrentamos a videos grabados con cámaras fijas que capturan movimientos de personas que, en función de la situación que ocurra, reclamarán en mayor o menor medida nuestra atención.

En pos de esto, surge un nuevo reto para la ciencia, conseguir reducir el espacio de almacenamiento y el tiempo de procesamiento, de tal forma que las imágenes captadas por los sistemas de videovigilancia mantengan la utilidad y mejoren su eficacia. Necesitamos tener sistemas rápidos que operen en tiempo real, de tal forma que su tiempo de respuesta sea mínimo y las autoridades de seguridad puedan actuar a tiempo ante las posibles anomalías causadas.

Nuestra propuesta en este proyecto es crear un sistema automático que apoye la videovigilancia en el transporte público, filtrando la información irrelevante y permitiendo a los sistemas de análisis posteriores, concentrarse en las áreas o regiones de mayor interés. Para ello, debemos seleccionar adecuadamente la tecnología a emplear: los métodos de clasificación tienen que ser eficaces y rápidos por la necesidad de responder en tiempo real, para ello, haremos uso de árboles de clasificación y *Random Forest*. Además, incluiremos en nuestro modelo características que nos permitan modelar aspectos de relevancia para nuestro ámbito. Teniendo particular cuidado con el tamaño de la muestra que manejamos ya que, al estar prediciendo la salida a partir de ella, debemos tratar de encontrar un modelo que se ajuste lo suficiente a los datos como para poder generalizar.

1.1. Motivación

La reacción ante los estímulos que poseemos los humanos ha sido siempre definida dentro del sistema nervioso, dominado por el cerebro, órgano al que llegan los estímulos y desde el que se generan las respuestas a estos, este sistema es el resultado de años de evolución, en los que se han ido matizando las partes esenciales de nuestra especie y que nos sitúan ahora en un escenario de investigación y desarrollo tecnológico que, más que nunca, se focaliza en conseguir imitar el comportamiento de la máquina más perfecta que jamás existió, nosotros mismos.[1]

Es la naturaleza de estas respuestas tan precisas y que tan poco tiempo nos supone tomar, la base de nuestro estudio, pues, en el plano visual, la atención es limitada. Esta limitación es la que hace que, a la hora de enfrentarnos a una imagen o un video vayamos directamente a las zonas que nos interesan, zonas en las que encontramos la información adecuada para la tarea a la que nos enfrentemos y, en caso de no tener ninguna tarea en concreto, zonas que nos resultan atractivas y de las que decimos que *"nos llaman la atención"*.

Ante el crecimiento de la información digital en la sociedad actual, el procesamiento de los datos generados se convierte en una tarea imposible de desarrollar por los humanos, teniendo que crear nuevos sistemas informáticos que nos permitan realizar estas tareas. Pero, aun así,

también estos sistemas tienen limitaciones a la hora de procesar la ingente cantidad de información, teniendo que recurrir a la ya famosa nube. En resumen, poseer un espacio de trabajo que nos permita almacenar y procesar datos hoy en día es asumir un conjunto de gastos en cuanto a tiempo, trabajo y dinero.

En este proyecto, nos concentraremos en el ámbito de la videovigilancia en el transporte público. En él, tenemos una tarea que desarrollar que conlleva el uso de múltiples cámaras que han de ser procesadas por operadores que, como hemos dicho, tendrán una atención visual limitada. Este hecho conlleva a tiempos de respuesta elevados que el análisis automático de las mismas podría reducir. Sin embargo, por otro lado, el hecho de contar con cantidades inmensas de información conllevaría que el tiempo de computo se viese incrementado, permitiendo así que algunos eventos no se detectasen, quedando perdidos en la gran cantidad de información captada. En este punto, los sistemas de atención visual permiten llevar a cabo un filtrado de tal información[2], facilitando la tarea a sistemas posteriores de análisis cuyo uso quedaría restringido a las áreas marcadas de interés por los primeros.

Se entiende así que nuestro modelo objetivo supondrá el primer eslabón de una cadena que nos permita dar un paso en el procesado de grandes masas de datos para conseguir una eficacia óptima de los mismos. Además, no podemos perder de vista que nuestra función también será desarrollar un modelo de respuesta rápido, por lo que emplearemos algoritmos de clasificación que cuenten, entre sus puntos fuertes, con un tiempo bajo de computación a la hora de clasificar cada dato que llegue. Los árboles de decisión y los *Random Forest*, pueden estar perfectamente catalogados como herramientas de clasificación de gran eficiencia computacional.

1.2. Contexto socioeconómico

Si nos centramos en el análisis y procesamiento de video, entraremos en una mina de contenido por explotar, ya que se prevé que el video represente el 80 % del contenido que circule a través de internet en el año 2019[3]. Resultará imposible para los sistemas conocidos hoy en día la anotación de estos videos y el procesamiento de los mismos; por lo tanto, nuevas técnicas han de ser investigadas y desarrolladas de tal forma que, una vez superemos el problema del almacenamiento de los datos, debamos enfrentarnos a la necesidad de emplear estas nuevas técnicas, siendo de especial interés en este campo los sistemas de atención visual basados en tareas que nos permitan filtrar la información no relevante o *distractora* de la verdaderamente relevante, que corresponde a lo que llamaremos fijaciones o *fixations*.

En el escenario sobre el que nos hallamos, debemos también reconocer cambios del contexto ya que, en la actualidad, la mayoría de medios de transporte incorporan a sus sistemas de seguridad circuitos de videovigilancia que generan video de manera continua y que es almacenado durante un tiempo a pesar de que pueden ser completamente prescindibles si solo contienen escenas en las que no ocurre nada. El análisis de estos videos mediante algoritmos que, incluso, puedan procesarlos en tiempo real supondría una doble ventaja ya que, por un lado, permitiría almacenar solo aquellos videos donde se detecten situaciones anómalas y, por otro lado, la seguridad aumentaría para la experiencia de los usuarios ya que sería posible prevenir posibles situaciones de riesgo. Esta ventaja supone por lo tanto, un aliciente a la hora de que las empresas de videovigilancia incorporen nuevos algoritmos a su línea y se incremente la investigación de estos modelos.

A la hora de valorar la calidad de un medio de transporte, uno de los aspectos cruciales es la seguridad de la que este dispone, tanto a la hora de efectuar el trayecto, como dentro de la

propia experiencia del viaje[4]. Es por esto, que hoy en día, resulta fundamental en todo medio de transporte que se precie, llevar equipamiento de seguridad que permita al usuario viajar con una base establecida de comodidad. Un circuito de videovigilancia puede ser suficiente para satisfacer esta experiencia y ayudar no solo a la seguridad de cada viajero, sino a la propia calidad de la empresa, ya que los servicios de seguridad no solo protegen a los usuarios, también evitan males mayores en las empresas a las que ofrecen su servicio.

Es por ello que, situarnos con un modelo que apoye la videovigilancia, puede garantizar una oportunidad de investigación ya que su uso es frecuente en la mayoría de servicios públicos y siempre se emplea con la misma función de seguridad. Es también parte de este estudio social el hecho de comentar que el modelo que intentaremos desarrollar no atenta contra la profesión que hoy en día regentan los vigilantes de cámaras de seguridad, sino que, su objetivo último es ayudarlos y servir como apoyo, proporcionándoles una herramienta capaz de filtrar su trabajo y llevar su atención a situaciones que probablemente requieran un mayor uso de su experiencia.

Por último, cabe destacar la importancia del video en la actualidad, ya que llevar a cabo un buen procesado del mismo, puede ser un factor diferenciador a la hora de evaluar a las empresas de cualquier entorno.

1.3. Regulación

Dado que nos estamos introduciendo en un tema que contiene una legislación muy en boga en la actualidad, es necesario realizar un análisis de los puntos claves de la misma. La llamada *Ley Orgánica de Protección de Datos de Carácter Personal*, fue expedida en el BOE del 14 de diciembre de 1999 y "tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar"[5].

En su ámbito de aplicación, artículo 2, para nuestro interés, encontramos que la ley se aplicará a los datos de carácter personal registrados en soporte físico para su tratamiento y uso en sectores públicos o privados: "Los procedentes de imágenes y sonidos obtenidos mediante la utilización de videocámaras por las Fuerzas y Cuerpos de Seguridad, de conformidad con la legislación sobre la materia."

Además, otro punto que afecta en este proyecto, debido a que no existe la posibilidad de acceder a base de datos captadas en medios de transporte cotidianos, se encuentra en el artículo 15.1, que establece el derecho de acceso a los mismos como: "El interesado tendrá derecho a solicitar y obtener gratuitamente información de sus datos de carácter personal sometidos a tratamiento, el origen de dichos datos, así como las comunicaciones realizadas o que se prevén hacer de los mismos". Por lo tanto, el acceso a los videos solo sería posible previo aviso de la empresa que gestiona los mismos a los organismos que regulan la ley, justificando su uso con fines de seguridad e investigación.

Por último, para matizar, debemos diferenciar que la ley se puede aplicar tanto al ámbito privado como público. Si nos encontramos ante empresas de transporte privadas, obtener los datos de videovigilancia supondrían un esfuerzo el doble de importante, ya que, además de la presente ley, sería necesario llegar a un acuerdo con la propia empresa.

1.4. Objetivos del proyecto

El objetivo principal de este proyecto es crear un modelo de atención visual dirigido por una tarea específica que sirva como apoyo en la videovigilancia de transportes públicos. Para ello,

estudiaremos distintos algoritmos y métricas que nos lleven a dar una mejor solución ante la base de datos que usaremos como fuente de información del proyecto.

Entre los objetivos particulares que encontramos realizando un análisis profundo, figuran:

- Crear una base de datos de fijaciones de la mirada de sujetos que etiqueten cada uno de los 30 videos empleados para el desarrollo.

- Analizar las fijaciones y diseñar las características idóneas sobre las que construir el modelo, optimizando estas de tal forma que su uso nos de las mejores prestaciones.

- Conseguir un modelo de clasificación que genere los mapas de saliencia que cumplan las expectativas para cada *frame*. Evaluar la capacidad de los árboles de decisión y *Random Forest* para generar modelos de atención visual *top-down*, ya que estos poseen la ventaja de contar con tiempos cortos de clasificación que nos vendrían bien a la hora de aplicar el modelo a situaciones de videovigilancia reales.

- Mejorar las prestaciones de modelos tradicionales de atención visual *bottom-up*, basados en la detección de estímulos salientes

- Tomar conciencia de las métricas de evaluación que se deben utilizar en problemas de este tipo.

Además, se puede incluir como objetivo previo al desarrollo del proyecto el siguiente punto:

- Conocer el funcionamiento y la configuración de la tecnología que nos permitirá la captación de datos de tal forma que lo podamos emplear con totales garantías para la consecución de los objetivos anteriores.

1.5. Metodología

La metodología que se ha seguido en este proyecto viene guiada por la ambición de desarrollar las cuestiones prácticas con un conocimiento teórico previo, de tal manera que conozcamos la naturaleza del problema para desarrollar una solución que se llevará a cabo empleando los siguientes pasos:

1. Leyendo y estudiando artículos relacionados con el problema en cuestión que nos permita realizar un esquema de los pasos a seguir.

2. Informándonos más profundamente de las técnicas que hay que llevar a cabo para resolver los distintos problemas, y aplicándolas en el diseño de nuestros algoritmos.

3. Implementando y testeando simulaciones que nos permitan obtener resultados para las soluciones propuestas en el punto anterior.

4. Comparando nuestros métodos y resultados con aquellos que consideremos son de un interés relevante para el problema y, volviendo a repetir los puntos dos y tres hasta obtener las prestaciones deseadas.

5. Cuando se requiere, se buscarán métricas que permitan comparar nuestros resultados e informarnos sobre los posibles errores que se estén cometiendo y los posibles puntos fuertes y diferenciales de nuestro desarrollo. De tal forma que se puedan revisar hasta quedar satisfechos.

6. Por último, se intentará referenciar y apoyar los resultados con herramientas gráficas que permitan dotar de una dimensión visual a los mismos.

Además, una vez se haya terminado de proponer la solución, se llevará a cabo un análisis de los resultados y una conclusión que ayude a enfocar el problema de forma global, de tal forma que se consiga realizar un proyecto coherente y cohesionado aportando una solución adecuada.

1.6. Estructura del documento

El presente proyecto está estructurado en seis capítulos y un anexo:

• Capítulo 1: Introducción del proyecto. Incluye una contextualización socioeconómica y la metodología empleada. La estructura, el marco regulador aplicado y la motivación del proyecto se encuentran en este.

• Capítulo 2: Estado del arte y tecnologías empleadas. Aparece aquí la descripción de los modelos de atención visual, así como una breve historia de los mismos, particularizando en el ámbito de la videovigilancia. Los algoritmos de clasificación son presentados en este capítulo.

• Capítulo 3: En este capítulo se desarrolla la creación de la base de datos, detallando los ficheros y videos que han sido utilizados para su creación. Distinguiremos en dos fases: configuración y captación.

• Capítulo 4: Desarrollo específico de los modelos de atención visual *top-down*. Se incluye en este capítulo la solución propuesta, así como la justificación de la elección de características empleadas y del clasificador utilizado.

• Capítulo 5: Experimentos realizados. En este capítulo se abordan los experimentos que han sido llevados a cabo para optimizar las características y el clasificador, asi como los resultados obtenidos en el mismo y la discusión de estos a través de comparativas con otros modelos anteriores.

• Capítulo 6: En este capítulo se aborda la conclusión global del proyecto y las líneas futuras que debería seguir esta investigación.

• Capítulo 7: Para finalizar, haremos un desglose a modo de estimación del presupuesto necesario para realizar este proyecto.

• Anexo I: El anexo incluye una amplia descripción de las métricas de evaluación sAUC y sNSS empleadas, además de la creación del *shuffle map*.

• Anexo II: En este apartado encontramos una descripción detallada del modelo GBVS propuesto por Harel y utilizado para la comparación de resultados.

• Anexo III: En él se incluye una descripción de las diferentes métricas para seleccionar los parámetros del clasificador.

• Anexo IV: El anexo incluye un resumen en inglés de partes fundamentales de este trabajo.

2. Estado del arte y tecnología aplicada

2.1. Estado del arte

2.1.1. Modelos de atención visual

En la sección anterior, se ha analizado la estructura de nuestro problema y la metodología que utilizaremos para enfrentarnos a él. Pero antes, es necesario analizar las numerosas investigaciones que han contribuido al desarrollo de los modelos de atención visual con la creación y la implementación de algoritmos que ayudan a la construcción de, cada vez, modelos más pioneros y con mejores resultados. No obstante, en la actualidad, seguimos dentro de una zona de constante cambio e investigación, en la que siguen apareciendo nuevos métodos y nuevas tecnologías que facilitan el progreso en este campo y que pasaremos a documentar en este apartado.

En primer lugar, debemos situar como punto de partida el artículo de Ali Borji de enero de 2013[6] en el que se realiza una de las revisiones más exhaustivas en cuanto al estado del arte de los modelos de atención visual. Haciendo un repaso de la historia de los mismos, encontramos:

- A inicios de la década de los 80, Treisman y Gelade proponen una nueva hipótesis sobre la atención que consiste en que cada estímulo está compuesto de una serie de combinaciones de características que se pueden separar para distinguir los posibles objetos. Encontrando formas y colores se pudo probar la hipótesis y con ello, se establecieron criterios que permitían hacer predicciones razonables en algunos casos y establecía los límites en otras. Su propuesta se conoce como "*Feature Integration Theory*"[7].

- En 1985, Koch y Ullman proponen que las diferentes características que componen un estímulo, forman un mapa topográfico direccional al que llaman mapa de saliencia y que sirve de medida de lo llamativas que son las regiones de las imágenes. En su estudio, muestran que la saliencia de una zona está determinada en primer lugar, por lo diferente que es la localización respecto a lo que rodea, basándose de nuevo en las características *bottom-up*. Introducen también el concepto de *winner takes all*, ya que establecen que la localización más saliente es aquella que centra la atención. El concepto de Koch y Ullman, sin embargo, no se verá implementado hasta 1996 con un trabajo de Niebur y el propio Koch[8].

- En 1998 Itti et al.[9] Realizan la primera implementación consistente del trabajo anterior y es aplicado a escenarios naturales, incluyendo en el trabajo la presencia de tareas de búsqueda. Con la creación de un software capaz de extraer las características de cada imagen y realizar una combinación lineal de las mismas, de tal forma que se pueda generar un mapa de saliencia, su trabajo favorece el interés en este campo.

- Además, con la llegada de tecnologías capaces de reconocer el movimiento de los ojos, nuevos modelos han surgido para tratar de describir matemáticamente los movimientos oculares que, deben estar ligados a la saliencia perceptiva y la atención. Algunos de estos modelos que se basan en los movimientos de los ojos son: "A*n ideal-observer model of reading*"[10]; el EMMA *model*[11] o el HMM[12] para construir modelos aleatorios de caminar basados en estadísticos de Markov.

En conclusión, tenemos dos líneas de avance que, hasta la actualidad, suponen las dos maneras de crear modelos de atención visual: aquellos que se basan en características *bottom-up* y aquellos otros que se basan en características *top-down*.

Para comprender la diferencia de ambos, nos situaremos como un sujeto x, que, desconociendo la naturaleza del problema, visualiza un conjunto de imágenes. En ellas encontrará diferentes zonas que le "llamen más la atención" que otras. Analizando las imágenes, encontraremos que la zona más llamativa, correspondía a una región que, como introduce Itti, es la más diferente en relación a las otras zonas y, por lo tanto, su vista se ha desplazado hacia ella en primer lugar. Para encontrar esta zona, tendremos que haber procesado la imagen de manera que extraigamos toda la información sobre colores, formas, texturas, orientación, etc. El modelo que infiramos de dicho experimento, será un modelo de atención visual *bottom-up*.

Si, por el contrario, nos situamos en nuevo escenario como un sujeto al que se le encarga realizar una tarea de búsqueda, como por ejemplo, encontrar aquellas imágenes en las que una persona esté tomando café, observaremos que también existen zonas en las que su atención se concentra por encima del resto. Pero, esta vez, cuando procesemos la imagen, no encontraremos que estas zonas se ajusten simplemente con las características anteriores, pues el usuario ya no se dejará llevar por los estímulos primario de la imagen, sino que buscará cosas tales como, el movimiento de los brazos, las tazas e, incluso, puede que en la imagen la persona que se tome el café esté de espaldas y sea imposible reconocer estas. El modelo, por lo tanto, será *top-down*, ya que para encontrar las zonas que atraen la atención, tendremos que entender primero la naturaleza de la tarea en cuestión. Nuestro modelo será de este tipo, ya que los usuarios detectaran situaciones anómalas en el transporte público. Un ejemplo de este tipo de algoritmos, que sirve como base a partir del cual generar nuevos algoritmos basados en características *top-down* es el de "T*op-down control of visual attention in object detection*", en el que se estudia como introducir información sobre el contexto de la escena a partir de los registros en la mirada de las personas[13].

La atención *top-down* es más lenta ya que está dirigida por alguna tarea, y no es involuntaria, como lo sería la atracción ante un estímulo. En 1967, Yarbus[14] pone uno de los ejemplos más famosos de este tipo de atención, demostrando que el movimiento de los ojos depende de la tarea en la que se encuentren a través del siguiente experimento:

Los usuarios verían la misma escena que consistía en una familia dentro de una habitación, en la que entraba un visitante, bajo diferentes premisas cada uno, como: estimar las circunstancias familiares, calcular la edad de cada miembro o simplemente analizar libremente la escena. El resultado del experimento, que se muestra en la siguiente imagen, muestra la escena original y los cuatro recorridos que hacen los ojos de los distintos usuarios. Así Yatbus consiguió demostrar que, en función de la tarea que se propusiese, las fijaciones, y, por lo tanto, el recorrido de los ojos, era diferente, como se puede observar en la figura 1.

Desde entonces, los modelos han explorado diferentes fuentes de influencia *top-down* que nos respondan la pregunta de cómo decidimos dónde miramos. Algunos modelos, presentan que, si nos proponemos seguir un objeto, las características de este serán las que atraigan nuestra visión y, por lo tanto, serán las que busquemos. Mientras que otros modelos investigan los diferentes contextos para averiguar qué localizaciones que atraen la mirada, ya que, ante tareas complejas, resulta difícil regular las fijaciones oculares. Se puede establecer, por lo tanto, que la atención *top-down* toma cada factor de forma conjunta ya que nuestra atención estará regulada por el diseño de la escena, así como del contexto que tengamos sobre la misma.

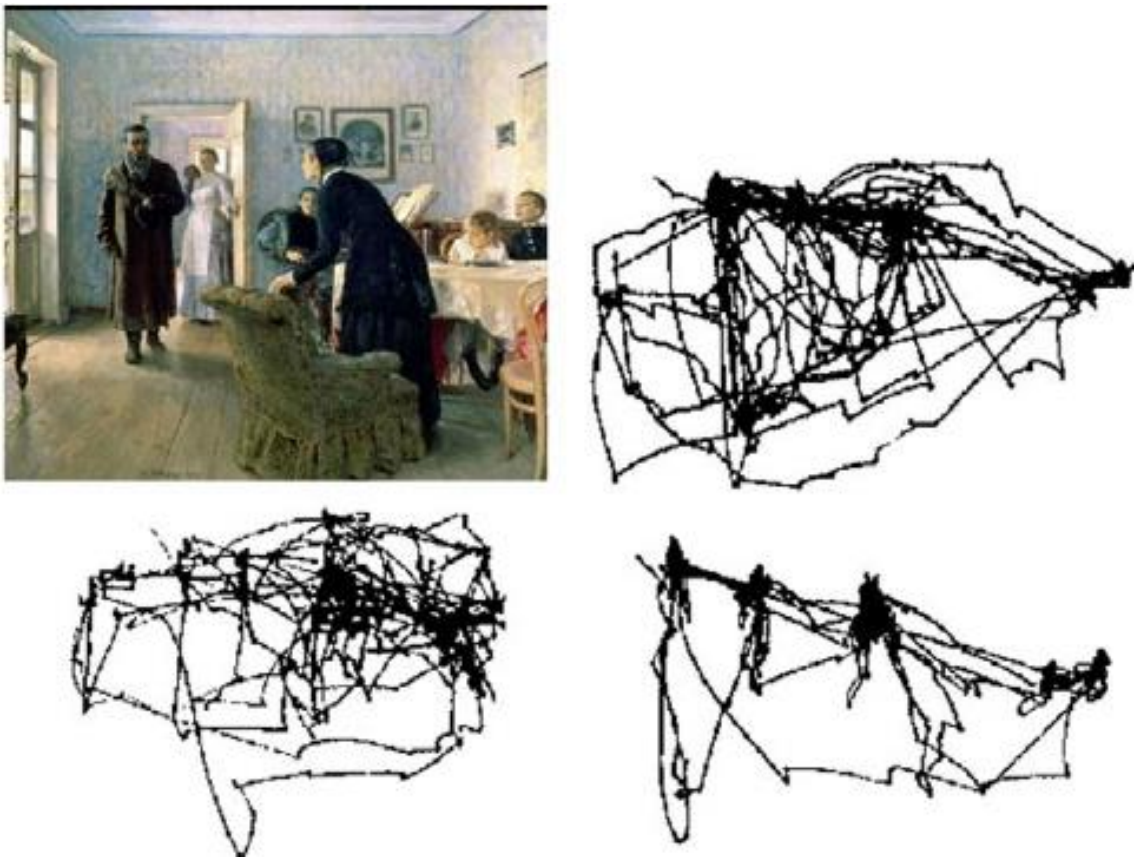


Figura 1. Ejemplo gráfico de los resultados del experimento de Yarbus, sacada de [14]

Por último, debemos señalar, que con la evolución y la entrada del *Deep learning* en la tecnología, se abre un nuevo y extenso mapa de ruta para las investigaciones de nuevos modelos[15] de atención visual, ya que, se está llegando a procesamientos en los que únicamente sea necesario introducir los datos al modelo, de tal forma que este encuentre de manera automática las mejores características y gestione de forma autónoma la red.

2.1.2. Sistemas automáticos de videovigilancia

En el ámbito de la videovigilancia, encontramos más algoritmos de detección, basados en detectar y seguir a personas o vehículos, que modelos de atención visual como tal. Aun así, dichos algoritmos están basados en un concepto totalmente correcto: si encontramos hacia donde se dirigen los humanos, tendremos los lugares hacia los cuales dirigen su atención y, por lo tanto, serán zonas similares y cercanas a las regiones de saliencia del video. Un ejemplo de estos modelos es el propuesto por Benfold y Reid, titulado "*Guiding Visual Surveillance by Tracking Human Attention*"[16]. Pero también podemos encontrar la aplicación de esta tecnología en este ámbito en el algoritmo descrito por María T. López[17] en el que describe un modelo basado en la descomposición de una imagen en otras más pequeñas que contienen peatones y vehículos y trata de predecir cuál es la más llamativa en cada momento. Otro modelo, quizá más parecido al propuesto por nosotros, es el introducido por Lucía Maddalena y Alfredo Petrosino[18] en el que basan el procesado en la detección de objetos en movimiento, utilizando el concepto de redes neuronales para clasificarlos.

Finalmente, podemos encontrar una revisión exhaustiva de esta rama de los modelos visuales basados en el movimiento de vehículos y humanos en el estudio de Weiming Hu [19] en el que se

analizan las distintas técnicas que existen de extracción de características y como se pueden utilizar y combinar las mismas para obtener los mejores modelos.

2.2. Tecnologías involucradas

2.2.1. *Eye tracker*

Una de las partes fundamentales que da vida a este proyecto consiste en el hardware utilizado para la captura de la mirada de los sujetos. Hemos utilizado un *eye tracker* remoto de la empresa SMI modelo RED. Además, se incorporaba la SDK *iView* que nos ha permitido comunicarnos con este, de tal forma que establecimos la conexión con el dispositivo a través de esta. Dadas las dos fases del proyecto, esta tecnología adquiere el papel protagonista en la primera de ellas, pues nos ha permitido obtener los datos finales de nuestra base de datos como mapas de *ground truth,* los cuales nos han permitido entrenar y evolucionar nuestro modelo.

Los tipos de datos con los que el *eye tracker* nos permite operar son:

- *Fixations*: Son los datos que nos interesan, consisten en coordenadas de determinadas localizaciones producidas cuando el *eye tracker*, en función de su configuración, establece que el usuario mira un punto concreto, con una desviación determinada.
- *Saccades*: Entre una fijación y otra, el ojo sigue un desplazamiento que puede durar varios *frames*, también depende su captación de la configuración del mismo. Estos datos los descartaremos y, en su lugar, interpolaremos las fijaciones para que se sostengan durante los *frames* y no tengamos *saccades*.
- *Blinks*: Son muestras que se toman como pestañeos, debido a que el *eye tracker* no puede captar el ojo y, por lo tanto, no puede hacer un seguimiento del mismo.
- *User events*: Estos últimos albergan los mensajes que manualmente introducimos para manejar los datos, ponemos uno cuando el video comienza a reproducirse y otro cuando acaba, para poder seleccionar temporalmente los datos que son propios del experimento.

Como principal característica de esta tecnología destaca su capacidad de trabajo a 250 Hz, permitiéndonos la captación de fijaciones y movimientos sacádicos en periodos variables de tiempo que podíamos fijar según interés. Además, la precisión que nos oferta la máquina permitía tener un control relajado de los usuarios a la hora de realizar la captación de datos, ya que, permite trabajar a una distancia entre 60 y 80 centímetros con movimientos de cabeza de hasta 25 cm/s de velocidad[20]. A continuación, en la figura 2, se muestra parte de las especificaciones del mismo:

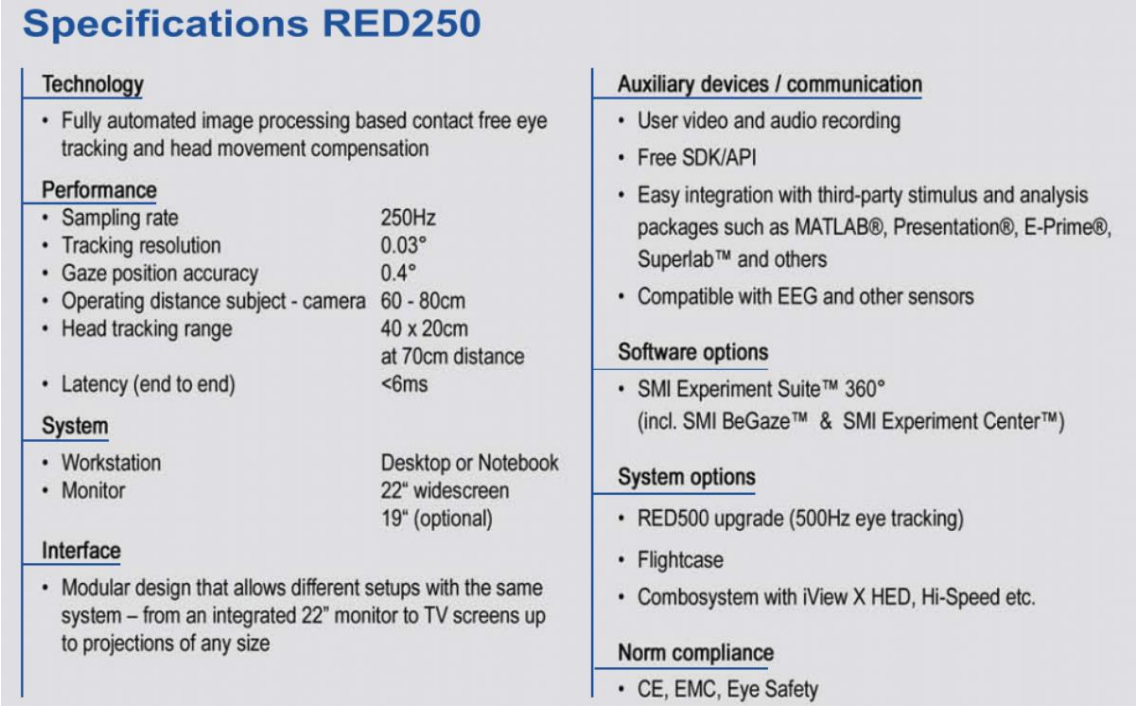


Figura 2. Captura de las especificaciones del Eye tracker

2.2.2. Árboles de clasificación

Para la segunda parte del proyecto, destaca como tecnología el algoritmo de árbol de clasificación[21]. Un árbol de clasificación se puede encontrar dentro de los métodos supervisados de aprendizaje automático y consiste en un conjunto de reglas de decisión que clasifican cada dato nuevo que llega a lo largo de un árbol hasta llegar a una hoja, que resuelva finalmente la clase a la que pertenece el dato. Los árboles se basan en una estructura ramificada que conecta el conjunto de nodos total del árbol, diferenciando tres tipos de nodos:

- Nodo raíz: En él se almacenan todos los datos sin dividir, es la puerta de acceso al dato nuevo que entra al modelo para ser clasificado
- Nodo intermedio: Son los nodos que forman el camino de decisiones que sigue un dato desde que entra en el nodo raíz hasta que llega a una hoja.
- Nodo hoja: Es el último paso que alcanza un dato a ser clasificado y en él se clasifica en última instancia el dato. El número total de nodos de este tipo marcará la complejidad del nodo.

El algoritmo de decisión está basado en diferentes reglas y criterios para seleccionar los nodos y las decisiones de cada uno. Estos criterios pueden ser diferentes y, por lo tanto, habrá que fijarlos en función del problema al que nos enfrentemos. Cuando se trata de crear un nodo, buscaremos aquella regla de separación tal que divida mejor nuestra muestra, discriminando las posibles clases de la misma. Es decir, si nuestros datos tienen 3 características o predictores (x1, x2 y x3) y dos clases (c1 y c2) buscaremos la regla que mejor divida esta, es decir:

Si haciendo a< x1 < b obtenemos que aquellos datos que cumplen la condición pertenecen el 50 % a la clase c1 y el otro 50 % a la clase c2, no estaremos haciendo una buena división, pues la condición no discrimina nuestros datos.

Si, por otro lado, cumpliendo la condición x2< c obtenemos que nuestros datos que la cumplen pertenecen el 95 % a la clase c1 y el otro 5% a la clase c2, habremos realizado una buena discriminación y por lo tanto la regla de decisión es mejor. Vemos este ejemplo en la figura 3.

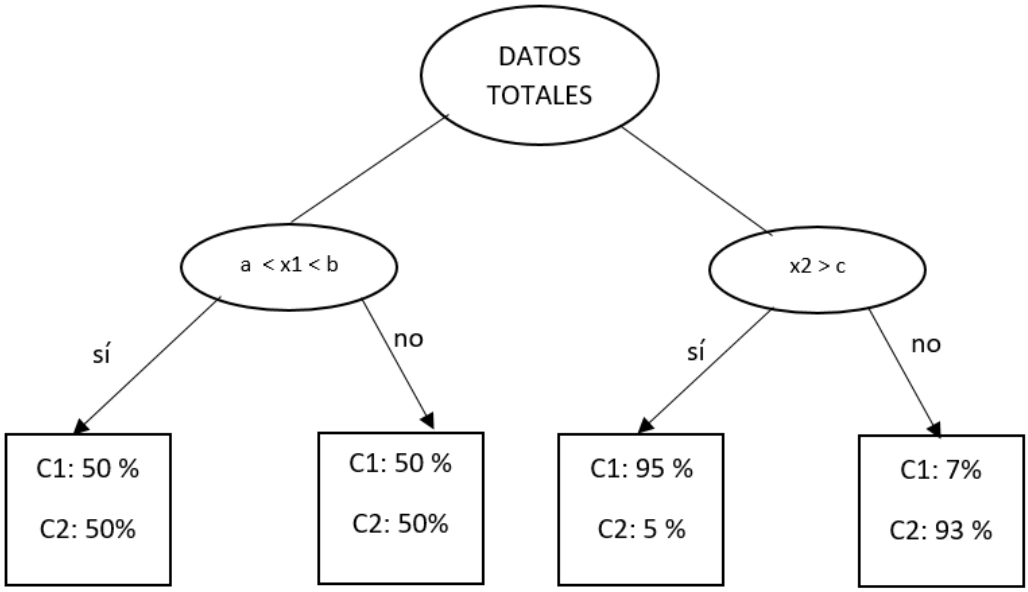


Figura 3. Ejemplo de árbol de clasificación

De esta forma que tenemos, por un lado, la capacidad de división de la muestra en cada nodo, y por otro, la cantidad de datos que se consiguen dividir. Ante esta causa, tendremos que definir un criterio de separación y un criterio de parada, es decir, cuando la división es tan óptima como para considerar que un nodo ha de ser un nodo hoja y no un nodo intermedio.

Finalmente, una vez que hayamos definido las reglas y criterios a emplear, la construcción del árbol se puede tratar de manera recursiva de tal forma que:

1. Seleccionamos el atributo que mejor divida los datos como nodo raíz y se crea una rama para cada posible valor.
2. Para cada nodo intermedio, el algoritmo vuelve a seleccionar la mejor regla que divida la muestra.
3. El siguiente paso consiste en crear las hojas del árbol si es posible con la división de los datos del nodo predecesor, según el criterio de parada, este se llevará a cabo o volverá al paso 2.

La idea global consiste, por lo tanto, en dividir el espacio de entrada recursivamente hasta llegar a nodos finales lo suficientemente puros, entendiendo por pureza del nodo, la capacidad de separación de este. De acuerdo con ello, debemos definir tres elementos para nuestro modelo:

1. ¿Cómo evaluamos la pureza de un conjunto de datos?
2. Una regla que nos permita seleccionar la mejor división de los datos, a la que llamaremos criterio de separación.
3. Una regla que nos permita determinar cuándo un nodo es hijo, a la que llamaremos criterio de parada.

Para evaluar la pureza de una muestra, podemos utilizar la medida de entropía, que nos dirá cómo de homogénea es la separación: si es total, la entropía del nodo toma valor 0. Es decir, la entropía es el valor del desorden de la muestra, en el anexo encontramos un desarrollo más extenso de la misma. Otra posible medida de esta, sería el propio error de clasificación del nodo, a la que también nos referiremos cuando desarrollemos el proceso de entrenamiento y validación de nuestro clasificador. Para seleccionar el criterio de separación de los datos o *split criterion* seguiremos las medidas de ganancia de información, que evalúa la diferencia entre la entropía de la muestra antes de la división y posterior a ella, y el índice de Gini[22], que se basa en la probabilidad de que un elemento sea bien etiquetado a través de relaciones frecuentistas entre el resto de elementos de la muestra, de esta segunda también encontramos un desarrollo en el anexo. Finalmente, para establecer un criterio de parada óptimo, estableceremos un conjunto de condiciones que actuarán como límites de crecimiento: estas pueden ser, un número máximo de niveles (profundidad), o, como será en nuestro caso, dejaremos crecer el árbol hasta el final y después podaremos el árbol en base al criterio CART[23], que combina el coste de predicción del árbol con la complejidad del mismo.

En conclusión, podemos resumir las ventajas y desventajas de los árboles en los siguientes puntos:

- Como ventajas destaca su alta capacidad de aprendizaje automático, ya que pueden manejar datos de diferentes tipos, numéricos o nominales e incluso datos de tipo *missing;* Sus prestaciones son buenas tanto para grandes conjuntos de datos como para muestras reducidas; y, por último, son fáciles de interpretar los resultados sin necesidad de conocer la ciencia que los precede.

- Entre sus desventajas podemos encontrar su alto riesgo de sobreajustar el modelo; su inestabilidad como clasificador, ya que pequeñas alteraciones en la muestra de entrenamiento pueden causar grandes cambios en cuanto a clasificación; suelen tener problemas respecto al sesgo debido a los valores que tomen las muestras, por eso que sea importante normalizar nuestras variables. Y, por último, podríamos generalizar que su capacidad de predicción no es tan óptima como la de otros clasificadores o modelos de regresión.

Por último, hemos de hacer un último análisis sobre la última fase que utilizaremos en nuestro clasificador, esta consistirá en utilizar un *Random Forest,* que no es otra cosa que utilizar un conjunto de árboles de clasificación de tal forma que se reduzca el sobreajuste del modelo con un único árbol. El algoritmo se basa en la técnica de muestreo *Bootstrap[240]*, que consiste en tomar la muestra original y generar, de forma aleatoria una colección de muestras a partir de ella. A la hora de generar esta segunda muestra, podremos seleccionar parámetros como el remplazamiento y el número de submuestras a generar.

Finalmente, cuando tengamos las nuevas muestras, el algoritmo las tomará como conjunto de entrenamiento y test internamente y generará un nuevo modelo. Para predecir la salida de este modelo, se empleará un sistema de votación como el que se muestra en la figura 4.

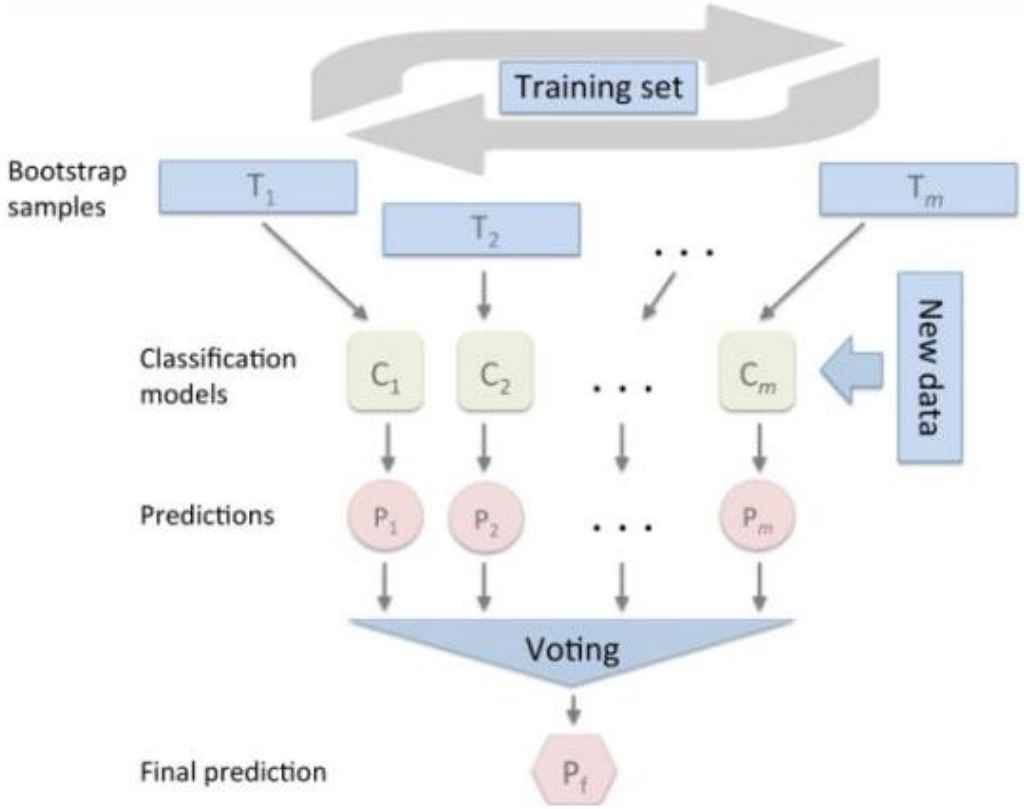


Figura 4. Ejemplo de entrenamiento de un conjunto de árboles o Random Forest

En la figura, cada sistema de clasificación sería un árbol de clasificación. La salida que utilizaremos no será exactamente el valor de la predicción ya que, esta nos diría si la muestra pertenece a una clase u otra, es decir, si la muestra es una *fixation* o no. Lo que vamos a utilizar es la probabilidad de esa muestra e pertenecer a cada clase, de tal forma que se termine generando un mapa de saliencia para la predicción de cada pixel.

3. Creación de una base de datos de video-vigilancia con fijaciones

La base de videos sobre la que trabajaremos para crear los *ground truths*, son un conjunto de 30 videos del proyecto BOSS del grupo Multitel[25] cuyas secuencias, de 2 minutos de duración media, simulan situaciones con diferentes anomalías dentro de un tren, alternando también con videos en los que no ocurre nada para dotar de más rigor nuestro proyecto. Además, las diferentes situaciones se graban desde 3 posiciones diferentes y, dado que 3 secuencias consecutivas tratan el mismo evento habrá que aleatorizar su visualización. El conjunto total de los videos lo podemos resumir en la tabla 1:

Evento del vídeo	**Número de secuencias**	**Descripción**
Robo de teléfono móvil	3	En estas secuencias, se simula un robo por parte de un viajero a otro del teléfono móvil.
Pelea	3	En este vídeo, se puede observar una pelea violenta entre dos hombres dentro del vagón
Enfermedad	3	Estas secuencias corresponden a situaciones en las que un viajero cae desplomado al suelo a causa de una enfermedad, y, el resto de ocupantes se dirigen hacia él.
Acoso de hombre a mujer	9	Este conjunto de videos corresponde a situaciones en las que un hombre acosa a una mujer, produciendo que la mujer abandone el sitio en el que se encuentra.
Discusión por el periódico	3	Estas secuencias simulan una discusión entre dos personas por el periódico.
Pánico	3	En estos videos podemos observar como actúa un grupo de viajeros ante un ataque de pánico producido por una alarma.
Ninguno	6	Estos videos no contienen ningún evento anómalo y son situaciones normales.

Tabla 1. Resumen y descripción de los videos de la base de datos

Incluimos, en la figura 5, un compendio de imágenes extraídas de la base de datos en las que se observan las distintas situaciones en las que existen anomalías y además, se muestran desde los distintos ángulos en los que están grabadas.

Figura 2. Ejemplos de frames extraidos de la base de datos

El procedimiento general para la creación de la base de datos, consiste en una toma de datos en bruto y, posteriormente, un procesado de los mismos para crear los *ground truth*. Por lo tanto, antes de captar los datos de los diferentes voluntarios, se tienen que realizar pruebas para garantizar que la toma de datos será correcta. Para ello, ilustraremos el proceso llevado a cabo y cómo podíamos evaluar nuestras facultades de cara a obtener éxito.

Lo primero que hicimos una vez terminada la configuración del *eye tracker* y la comprobación de la correcta extracción de las fijaciones suministradas por él, fue llevar a cabo la visualización de los videos y la toma de los datos por parte de los 15 voluntarios.

Nuestro objetivo último es conseguir etiquetar las fijaciones de todos los *frames* que hemos utilizado en nuestro experimento, de un total de 30 videos. Para ello, como se ha comentado en el apartado de tecnologías, empleamos la SDK de *iView*, que nos permitió la intercomunicación con el *eye tracker*. Dividimos el proceso de creación de la base de datos en dos fases, una primera para la configuración del software pertinente y otra para la captación final de los datos.

3.1. Fase de configuración

En primer lugar, tenemos que conseguir sincronizar la reproducción del video y la captación de datos del *eye tracker*. Para ello, nos apoyaremos en los mensajes de usuario que la aplicación nos permite añadir al conjunto de los datos. Una vez han sido captados, es necesario realizar un estudio de los mismos ya que encontramos distintos tipos dentro del documento generado. Cada dato viene etiquetado con el tiempo de inicio en el que ha sido producido, la duración total, y las coordenadas espaciales, si es posible, registrada. Como se ha explicado en el apartado dedicado al *eye tracker* en el apartado anterior, contamos con cuatro tipos diferentes de datos, las *fixations*, las *saccades*, los *blinks* y los mensajes de usuario.

En la figura 6 vemos una captura de la cabecera de uno de estos archivos generados:

```
Version:        IDF Event Detector 3.0.20
Sample Rate:    60
Subject:        User1
Description:    Description1

Table Header for Fixations:
Event Type      Trial   Number  Start   End     Duration        Location X      Location Y      Dispersion
X       Dispersion Y    Plane   Avg. Pupil Size X       Avg Pupil Size Y

Table Header for Saccades:
Event Type      Trial   Number  Start   End     Duration        Start Loc.X     Start Loc.Y     End Loc.X
        End Loc.Y       Amplitude       Peak Speed      Peak Speed At   Average Speed   Peak Accel.
Peak Decel.     Average Accel.

Table Header for Blinks:
Event Type      Trial   Number  Start   End     Duration

Table Header for User Events:
Event Type      Trial   Number  Start   Description

Table Header for Trigger Line Events:
Event Type      Trial   Number  Start   Duration        Port Status

UserEvent       1       1       6348395746      # Message: inicio video
UserEvent       1       2       6348578850      # Message: fin del play()
UserEvent       1       3       6455509746      # Message: fin del video
Fixation L      1       1       6344538728      6344622136      83408   1373.25 813.99  14      6       -1
        14.85   14.85
Saccade L       1       1       6344622136      6344655499      33363   1364.37 812.88  796.93  728.80
3.54    275.67  0.50    106.20  8068.10 -7877.28        7364.33
Fixation L      1       2       6344655499      6344838998      183499  806.01  704.77  12      32      -1
```

Figura 6. Captura de un fichero tipo de nuestra base de datos

Una vez tenemos, pasaremos a realizar una lectura de estos hasta que podamos tenerlos dispuestos de tal forma que seamos capaces de visualizar un video de ejemplo con las fijaciones sobreimpuestas. Cuando conseguimos que el seguimiento se complete, pasaremos a la fase de captación de datos entre personas voluntarias que se desplazaran hasta el lugar donde se encuentra el *eye tracker*, en la Universidad Carlos III de Madrid.

3.2. Fase de captación

Los participantes en el estudio son un total de 15 personas, entre 20 y 30 años de edad y de diferente género. Para empezar, se sitúan frente al ordenador, a una distancia controlada por el software del *eye tracker* y con la mirada centrada en el video en cuestión. El video se mostrará sin distorsión alguna, utilizando la *image toolbox* de MatLab, centrado en la pantalla y con un fondo de color gris uniforme a su alrededor.

Cada usuario está sentado durante un máximo de 30 minutos en el que verá un total de 10 secuencias de video distribuidas aleatoriamente para cada participante. Previamente, el sujeto será informado de que debe observar el video con el objetivo de detectar situaciones anómalas en los mismos, aunque sin la necesidad de comentar ni tomar nota de las mismas. Además, el hecho de tener diferentes ángulos de cámara para los mismos videos, nos ayudará a estudiar las zonas más atencionales, ya que, en ambos videos estas han de ser las mismas. La captación y vista de vídeos la dividiremos en dos días, de una duración aproximada de 4 horas cada uno de ellos, con 7 personas el primer día y 8 el siguiente. Así hasta completar el estudio y tener cinco ficheros de datos para cada video.

Finalmente, para completar la base de datos, debemos extraer las fijaciones y otros movimientos de los 150 archivos generados. Una vez que los movimientos sacádicos son descartados, nos quedamos con una serie de fijaciones que pueden no ser suficientes para todos los *frames*, por ello, se lleva a cabo una interpolación temporal de las mismas de tal forma que ningún *frame* quede sin fijaciones y así obtenemos una información más robusta, podemos visualizar el resultado como el ejemplo de la figura 7.

Para ello, llevamos a cabo un proceso de interpolación lineal del tipo vecino más cercano[26] que colocará la fijación a interpolar en el mismo sitio en el que se encuentre la *fixation* más cercana en el tiempo.

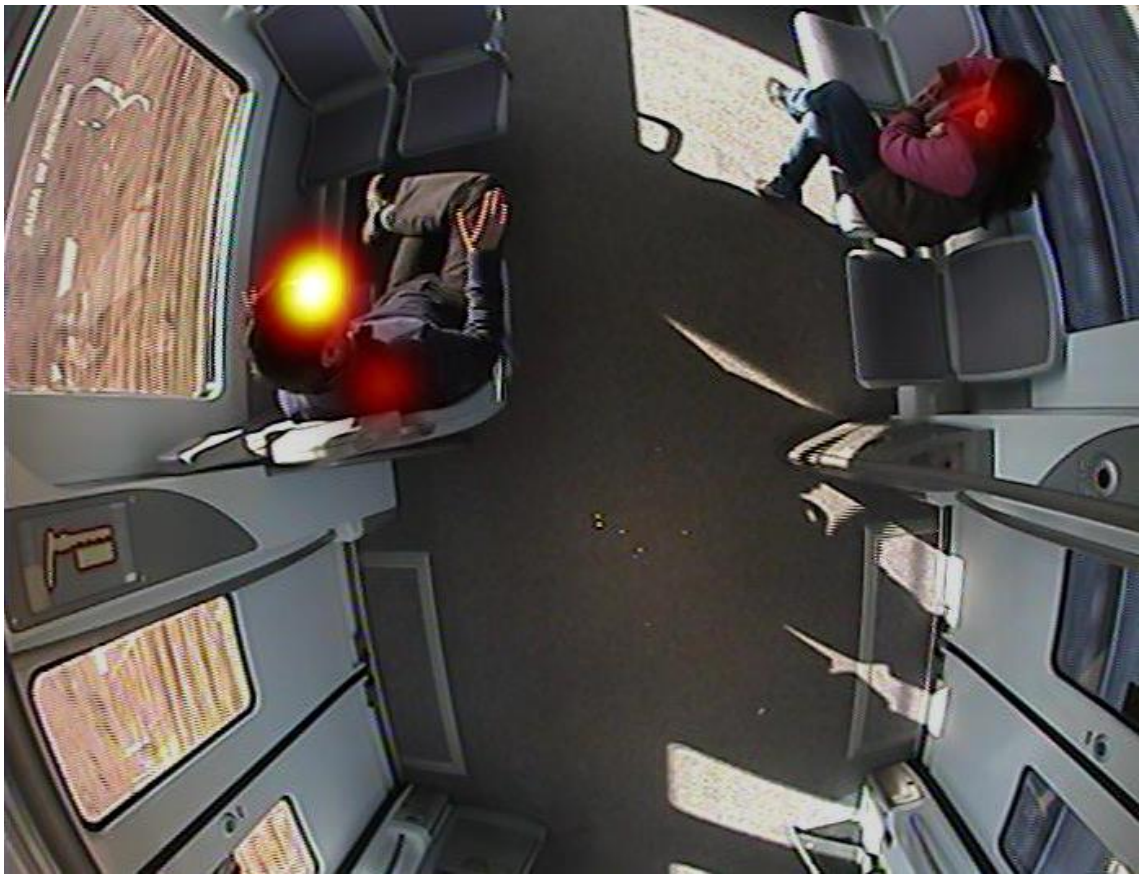


Figura 7. Ejemplo de frame con fijaciones sobreimpuestas

4. Solución propuesta: Modelo *top-down* de atención visual

El modelo propuesto está compuesto por tres procesos en serie que, a su vez pueden estar divididos en otra serie de subprocesos realizados en paralelo para, finalmente, aportar un resultado final en forma de mapa de atención visual. En primer lugar, la entrada del sistema consta del *frame* a generar el mapa de atención en cuestión, junto con el *frame* correspondiente anterior, si lo hubiera, para extraer sus características dinámicas. Una vez hemos realizado, en la primera etapa, la extracción global de características, obtenemos a la salida un conjunto total de 8 imágenes que debemos muestrear para pasar al clasificador y obtener el mapa final.

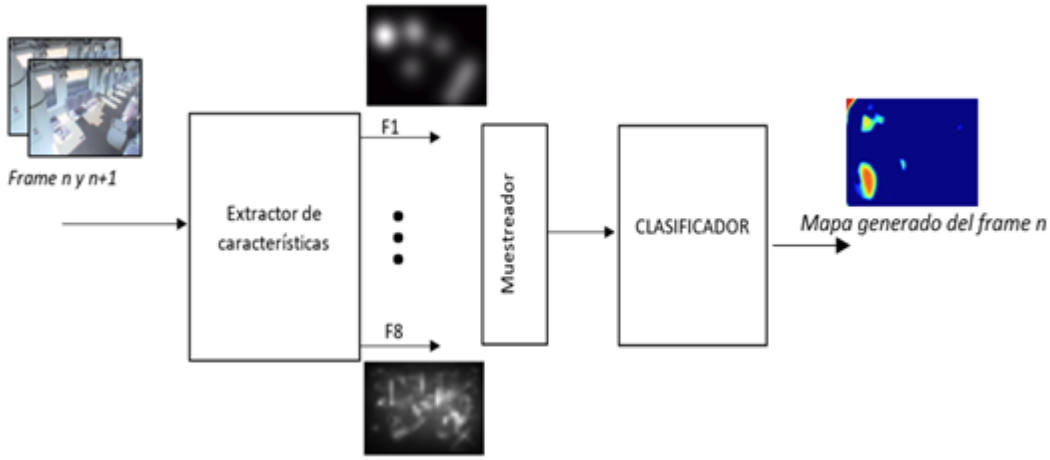


Figura 8. Diagrama de bloques del modelo propuesto

El esquema de la figura 8, en conclusión, queda definido en tres etapas:

1. Extracción de características: En esta etapa tenemos como entrada dos imágenes y como salida 8 mapas en escala de gris correspondientes a cada característica.

2. Muestreo espacial: El muestreador elegirá los puntos de las imágenes procedentes del extractor de características y generará una tabla con los valores de cada característica en cada punto muestreado. A su vez, tendrá dos comportamientos diferentes, uno para *train* y otro para *test.*

3. Clasificador: Con los datos que salen del muestreador predice una salida que clasifica cada punto según a la clase que pertenece, siendo 1, aquellos que corresponden a una zona de *fixation* y 0 los que no. Para generar el mapa suavizado de la salida, se basa en la probabilidad de pertenencia a la clase uno de cada punto.

4.1. Características

A la hora de evaluar las características que sirvan para obtener el modelo, nos basaremos en el diseño de la escena y en el contexto de nuestra tarea. Buscaremos obtener de cada escena las características que mejor modelen la tarea propuesta. Como apoyo, vamos a visualizar las fijaciones extraídas en la base de datos superpuestas en los videos de tal forma que podamos hacer una lluvia de ideas sobre cuáles pueden ser las zonas que llaman la atención de los sujetos del estudio, intentando extrapolar y generalizar para obtener las *features* que harán nuestro modelo próspero. Las características utilizadas las podemos dividir en dos grupos: El primero corresponde a características de tipo *bottom-up* genéricas que son extraídas con el modelo

GBVS[27], del que encontramos un desarrollo en el anexo. En el segundo grupo, particular para la tarea en cuestión, encontraremos características que consisten en el movimiento de la imagen y la detección de pasajeros. Además, introducimos el sesgo central de la visión de forma que se recoja la centralidad de las fijaciones en media. La lluvia de ideas nos propone una serie de puntos que debemos analizar para conseguir los descriptores:

- El movimiento de los pasajeros en el tren es causa de fijación ante situaciones como, por ejemplo, un tren vacío o pasajeros sentados cuasi-estáticos. Para ello, tendremos que estudiar el flujo óptico para encontrar las zonas dónde se produzcan movimientos. La naturaleza de los videos nos crea un problema a través de las ventanas, ya que los cambios de luz supondrán motivo de movimiento y no son zonas de atención visual, por lo que tendremos que estudiar como eliminarlas.

- La propia aceleración del movimiento causa fijaciones ya que cuando un pasajero se mueve "bruscamente" es una buena fuente de información ya que estos movimientos pueden derivar con frecuencia en situaciones anómalas (robos, desmayos, agresiones, etc…) aunque, también podría ser motivo de un cambio de iluminación total como, por ejemplo, la salida de un túnel.

- Las fijaciones se producen de media en el centro de la imagen, por lo que será conveniente estudiar el *center-bias* como característica. No obstante, nuestra base de datos contiene videos con los pasajeros sentados en los asientos del tren, que están situados en la parte lateral y, por lo tanto, la acción, no siempre se producirá centrada.

- Las caras de los pasajeros suponen un motivo claro de fijación, de hecho, cuando se localiza a un pasajero, la atención se llevará siempre a la zona facial, pudiendo ir a otros puntos según el movimiento de estos, como las manos y las piernas.

Tras estas ideas, debemos darles cuerpo de tal forma que encontremos descriptores de las imágenes que las modelen. En primer lugar, tendremos que utilizar un algoritmo de estimación de movimiento, tanto para calcular la magnitud del movimiento, como para calcular la propia aceleración del mismo. En segundo lugar, también tendremos que implementar un detector de caras o de pasajeros que nos proporcione buenos resultados.

4.1.1. Características de movimiento: Magnitud y aceleración

Para la aceleración y la magnitud del movimiento, emplearemos el algoritmo propuesto por Farneback[28], que consiste en la expansión de polinomios. El método empieza por aproximar vecindarios de un píxel en dos *frames* consecutivos a través de expresiones polinómicas cuadráticas. Para lograrlo, se utiliza la transformación de expansión polinómica y se evalúa como cambian estos polinomios en ambos *frames*, para, en función del valor que tomen sus pesos o coeficientes, dar un resultado vectorial (magnitud y dirección) sobre el movimiento que sufren los píxeles. El algoritmo incorpora un conocimiento a priori ya que asume que las coordenadas de un vecindario no varían mucho de un *frame* a otro y, por lo tanto, encontrar el mismo vecindario en dos *frames* consecutivos no es difícil. Pero, esta situación puede resultar problemática en largos desplazamientos y, ante esto, el movimiento tiene que ser aproximado a priori para poder estudiar los polinomios en coordenadas distintas.

El descriptor de la magnitud lo modelaremos directamente como la magnitud del vector de movimiento:

$$Magnitud(pixel_i) = \sqrt{x_i^2 + y_i^2}$$

Ecuación 1. Cálculo analítico de la magnitude de movimiento

Gráficamente resulta fácil interpretar esta característica visualizando cada *frame* y su magnitud de movimiento asociada correspondiente, teniendo en cuenta que nuestro algoritmo se basa en la diferencia entre el frame actual y el anterior, así, podemos observar lo siguiente:

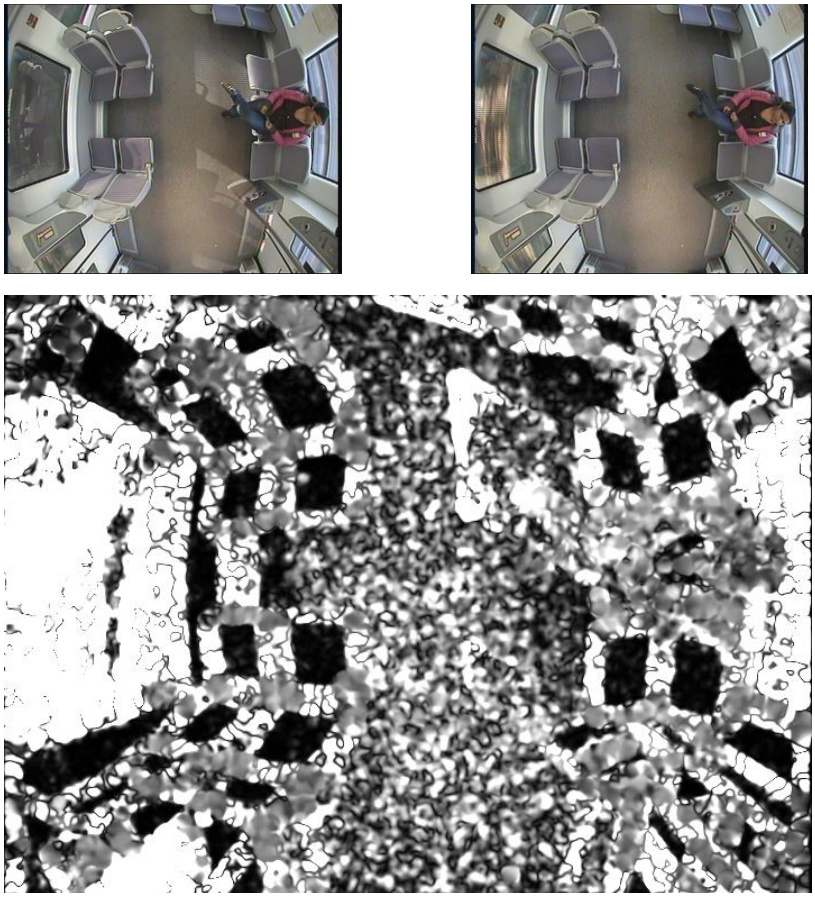


Figura 9. Ejemplo de la extracción de la magnitud a partir de dos frames

Podemos apreciar que en las zonas donde encontramos mayor magnitud de movimiento, se debe a cambios de iluminación y, en los casos en los que existe un número alto de pasajeros, por los propios movimientos de estos.

El descriptor de imagen de aceleración está basado en localizar movimientos rápidos que supongan pasar de un estado estático (pasajero sentado) a otro de movilidad (pasajero se levanta y corre) de tal forma que la acción implique atención visual. Para conseguirlo, debemos trabajar de nuevo con el método de estimación de movimiento de Farneback ya que es necesario procesar las componentes x e y del vector de movimiento.

Como estamos ante una sucesión discreta de planos, la aceleración pasará de ser la derivada de la velocidad (característica magnitud) a ser el módulo de la diferencia de componentes[29]. Es decir, no podemos perder la componente direccional ya que el movimiento es necesario captarlo en cualquier dirección. Analíticamente, calcularemos el valor de cada píxel del mapa final como:

$$Vx = Vx(n) - Vx(n-1)$$

$$Vy = Vy(n) - Vy(n-1)$$

$$Aceleracion(pixel) = sqrt(Vx^2 + Vy^2)$$

Ecuación 2. Cálculo analítico de la magnitude de aceleración

Por último, el resultado final se calcula como la norma de ambas componentes y se guarda en una nueva imagen generada de la misma manera que la del descriptor de magnitud:

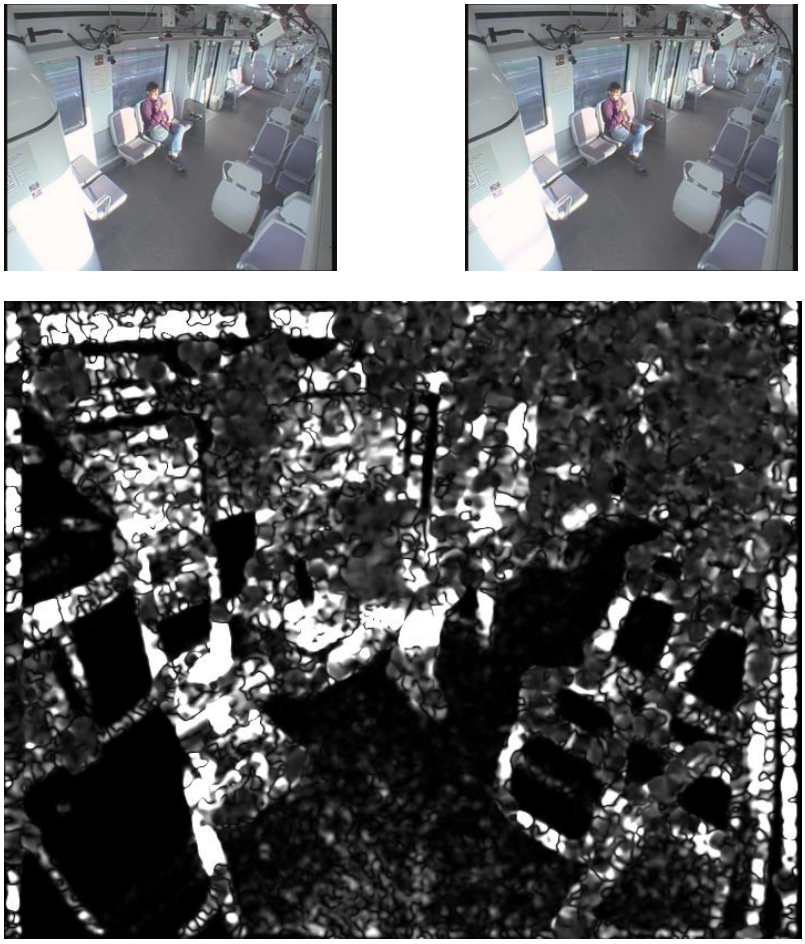


Figura 10. Ejemplo de extracción de la característica aceleración

4.1.2. *Center-bias* y *shuffle map*

Tanto el *center-bias*[30] como el s*huffle map* serán las características que emplearemos para modelar el sesgo de centralidad de la visión humana.

El *center-bias*, consiste en la tendencia natural de los observadores a fijar la atención en el centro de la imagen tanto cuando se visualiza una imagen en una pantalla, como en escenarios naturales. Además, en nuestro caso, siempre que aparezca un nuevo video, los usuarios tenderán a situar la atención en el centro de la imagen. Biológicamente, esto puede tener diferentes explicaciones:

- El centro de la imagen puede ser la localización óptima para procesar el resto de la escena.

- El centro de la imagen puede ser el mejor lugar de partida para explorar la imagen.
- Por último, puede que simplemente se trate de una tendencia natural de centrar la visión y recolocar al ojo en su órbita.

En nuestro caso de aplicación, esta característica no debe afectar mucho a la hora de predecir los mapas de saliencia ya que los videos se toman con cámaras estáticas en las que la anomalía no tiene porqué transcurrir en el centro de la imagen. Pero, teniendo en cuenta que, en última instancia, una persona tiene que ver los vídeos, la tendremos en cuenta.

La técnica de este descriptor que hemos de utilizar se desarrolla en el apartado de experimentos dedicado a la optimización de características, para la cual utilizaremos una gaussiana centrada en el centro de la imagen.

Por otro lado, también se incluye el *shuffle map* como característica. La explicación y desarrollo de cómo se obtiene viene en el anexo de este proyecto. En líneas generales, el *shuffle map* será tratado como una densidad de probabilidad de las localizaciones que ocupan las fijaciones en nuestro caso particular. Por lo tanto, para tener un modelo que corresponda a nuestra base de datos debemos incluir este, pero, en cambio, si queremos un modelo que generalice, será importante introducir un modelo de *center-bias* como el que se muestra en el experimento.

4.1.3. Detección de pasajeros

Otra característica que se podría considerar de gran valor de atención es la detección de los pasajeros. Quizá sea esta la más evidente ya que la saliencia que tiene el rostro humano y su cuerpo es, posiblemente, de las más llamativas en este tipo de estudios. No obstante, al enfrentarnos a videos de cámaras de seguridad en los que las caras no están muy definidas por la propia resolución de la imagen y el tamaño de las mismas, detectar bien las caras es muy problemático y no arroja buenos resultados.

Para conseguir una detección que podamos emplear con ciertas garantías, utilizaremos un detector basado en movimiento de objetos que además nos proporciona un seguimiento de los mismos. El algoritmo está basado en dos puntos diferentes, en primer lugar, la detección de objetos en movimiento utiliza un algoritmo de sustracción basado en modelos de mezcla de gaussianas, que proporciona una máscara a la que, mediante aplicaciones morfológicas, se consigue eliminar el ruido y, finalmente, se utiliza un análisis de regiones [31] en cada grupo de píxeles conectados de tal forma que cada grupo se pueda asociar a un objeto. En segundo lugar, la detección del objeto conlleva que se asocie al mismo objeto, si así ocurre. Esta asociación se lleva a cabo a partir del movimiento del conjunto de píxeles y se utilizan filtros de Kalman, de tal forma que se predice la localización del objeto y se determina el parecido a la detección anterior. El filtrado de Kalman[32] consiste en una herramienta matemática que se basa en la probabilidad que tiene un conjunto de sufrir una transformación, estudiando la covarianza entre la posición y la velocidad, de tal forma que nos proporciona un resultado robusto para predecir movimiento en grupos de píxeles.

Además, el algoritmo de detección basado en movimiento, nos va a permitir discriminar unos movimientos respecto a otros ya que, cuando se detecten objetos que no pueden ser *trackeados* durante un número determinados de *frames* los podremos eliminar. Esto supone que tenemos que asumir que un pasajero se mantendrá "en escena", durante un número mínimo de *frames*, de tal forma que, si no lo cumple, la detección será descartada y así eliminaremos movimientos provocados, por ejemplo, por cambios de luz.

Está característica nos puede resultar clave a la hora de desarrollar un buen modelo ya que, el detectar o no un pasajero puede suponer una gran diferencia de atención visual. A la hora de realizar los experimentos pertinentes, será esta en la característica que pongamos más énfasis debido a su enorme abanico de posibilidades ya que, a la hora de evaluar un pasajero, podemos pensar que alguna parte del mismo será más saliente que el resto, aunque esto no siempre es cierto ya que, por ejemplo, la acción puede estar desarrollándose con las manos y estas atraigan más nuestra atención. En el capítulo de experimentos, trataremos de encontrar la mejor solución a ello.

4.1.4. Características *bottom-up* genéricas

El algoritmo de saliencia visual basado en grafos y conocido como GBVS nos proporciona tres mapas de características *bottom-up* que incorporaremos a nuestro estudio debido a su gran consistencia y su buen aporte de información. En el anexo, encontramos un desarrollo de los pasos fundamentales del algoritmo. Como parámetro de entrada únicamente basta con introducir una imagen y el propio algoritmo nos ofrecerá un mapa de saliencia aproximado y una serie de características intrínsecas del propio modelo. Para nuestro estudio, incorporaremos las que corresponden a los *top level feat maps* que corresponde a tres mapas de características *bottom-up* de una resolución inferior que el *frame* original. Un ejemplo de estos mapas sería el mostrado en la figura 10:

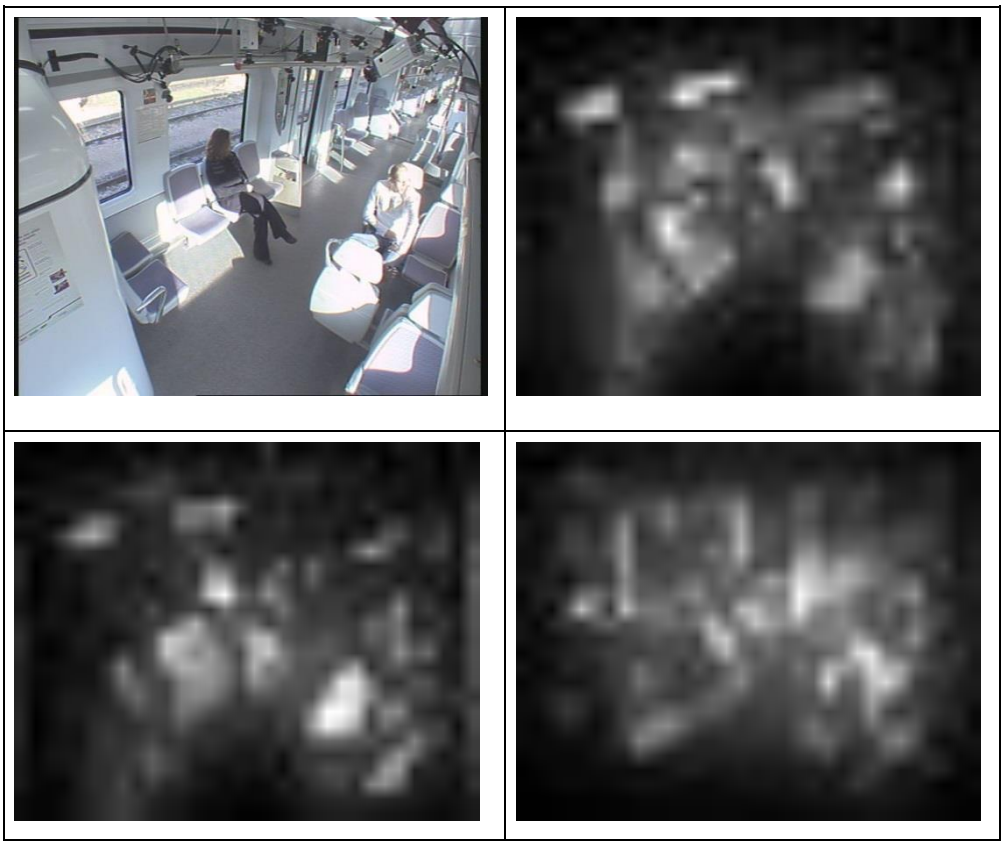


Figura 11. Ejemplo de mapas de características bottom-up

4.2. Muestreo

En nuestro modelo hemos empleado dos tipos de muestreo diferente, dependiendo de si estábamos en la etapa de *train* o validación y otro para validar *features* y realizar el test final del sistema. Estos dos procesos han sido:

- Probabilístico: Es el muestreo empleado en la etapa de entrenamiento y validación, el proceso seguido es el desarrollado en este apartado.
- Uniforme: Es nuestro segundo tipo de muestreo empleado. En él se reduce la complejidad y se busca optimizar el tiempo de computación. Este segundo tipo solo se utiliza para realizar el test final y consiste en seleccionar muestras aleatorias de uno de cada diez *frames* de nuestra base de datos. En cada *frame* hemos seleccionado 200 puntos.

Para seleccionar los puntos que utilizaremos para crear la matriz de datos de entrenamiento y validación (tipo de muestreo probabilístico), debemos utilizar un muestreo acorde a la naturaleza del problema, pues, las fijaciones están correspondidas a puntos concretos, pero, la zona que rodea a este no se puede tratar como un negativo de saliencia y por lo tanto, habrá que tener en cuenta esta consideración a la hora de evaluar positivos y negativos. Este proceso, se puede observar de manera gráfica siguiendo los siguientes pasos:

1. Creamos un espacio de guarda donde están situadas las fijaciones utilizando una desviación de valor 8, como vemos en la figura 12.

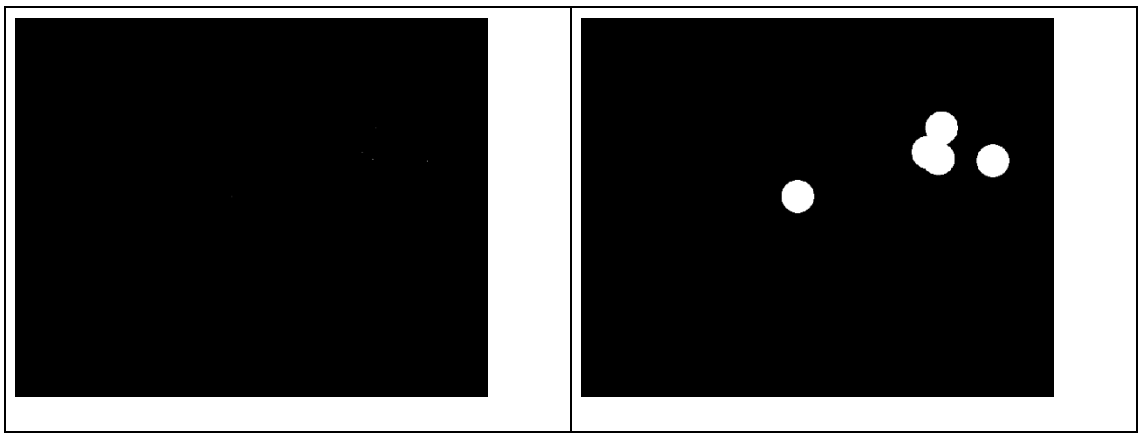


Figura 12. A la izquierda, mapa con las fijaciones, a la derecha, el mismo mapa, pero con estas ampliadas

2. Utilizamos la imagen obtenida para establecer la zona donde muestrearemos los negativos, para ello, y que no sea al azar, utilizamos el *shuffle map*, que contiene la probabilidad global de las fijaciones del experimento y del cual encontramos el desarrollo en el anexo. Si eliminamos las zonas donde hay fijaciones (positivos) del *shuffle map*, nos queda la probabilidad de muestreo de cada punto:

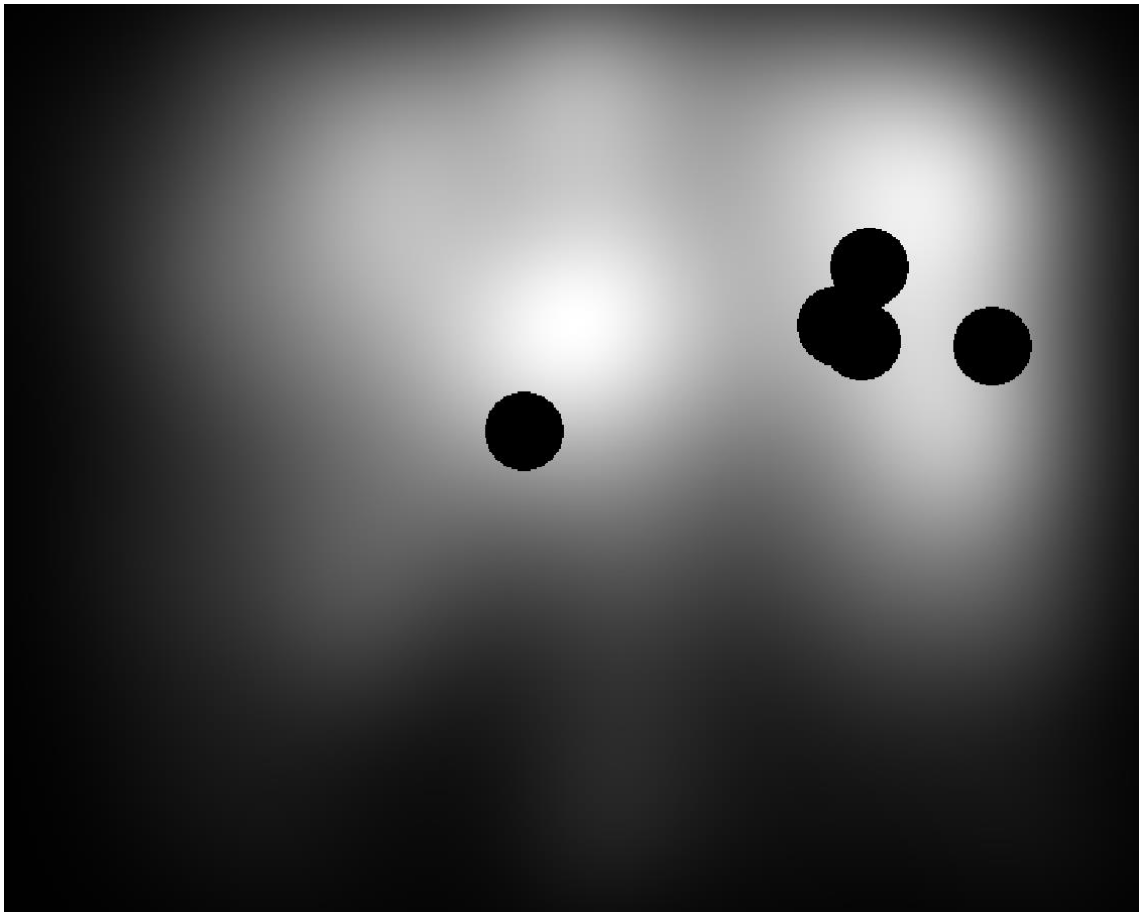


Figura 13. Zona de muestreo de los datos negativos

3. Tratando la figura 13 como una densidad de probabilidad, tomaremos cinco puntos por cada *frame* siguiendo la probabilidad de cada punto.

Así, finalmente, conseguimos un conjunto de diez coordenadas por cada *frame*, de las cuales, cinco pertenecen a las fijaciones de cada usuario que visualizó el video y, las otras cinco, pertenecen a cinco negativos muestreados de la forma cuasi aleatoria detallada en los 3 pasos anteriores. El uso del *shuffle map* nos permite separar las zonas próximas a las fijaciones de tal forma que no se tomen estas como negativos. Visualmente podemos pensar en un rostro humano en el que la fijación se encuentre puntualmente en la nariz del individuo, si no aislamos la cara entera, podríamos establecer que el resto del rostro no requiere la atención y por lo tanto, no estaríamos haciendo una revisión real. Por ello, es necesario tener esta consideración con el espacio que rodea la *fixation.*

El tamaño de las muestras que hemos utilizado han sido de 10 puntos por *frame*, en 100 *frames* aleatorios de 20 videos también elegidos aleatoriamente, para fijar los parámetros del clasificador, intentando evitar, en la medida de lo posible el sobreajuste. Mientras que, para entrenar el clasificador final, se han utilizado 10 puntos por cada *frame*, tomando 200 *frames* de cada video de la base de vídeos total.

4.3. Clasificador

Como ya hemos visto en el apartado de tecnologías empleadas, el clasificador que vamos a utilizar consiste en un *Random Forest* que aplica y combina diferentes árboles de clasificación. Particularizando para nuestro caso, utilizaremos una clasificación basada en 8 características,

correspondientes a cada uno de los descriptores de las imágenes que trataremos de forma normalizada entre 0 y 1, puesto que no hay ninguno que consideremos que tenga que tener mayor valor que otro. La fase de entrenamiento la haremos de forma balanceada, es decir, vamos a introducir tantas muestras positivas como negativas a las que tendremos que clasificar en dos clases diferentes. El proceso que llevaremos a cabo para diseñar el clasificador óptimo, paso a paso, es el siguiente:

- En primer lugar, debemos determinar los parámetros óptimos de un solo árbol de clasificación que mejor se ajuste a nuestro problema. La muestra que utilizaremos para ello será pequeña, ya que es muy fácil que cometamos sobre ajuste y, por lo tanto, será mejor emplear pocas muestras y una técnica *k-fold*. Para realizar el testeo de los parámetros, utilizaremos como métricas de calidad, la tasa de acierto, error y margen, que consisten en evaluar el porcentaje de muestras bien clasificadas, el error que se comete de media y la separación media entre muestras de distintas clases, respectivamente.

- Una vez que el árbol esté listo, pasaremos a diseñar una "bolsa" de árboles o *Random Forest*, al que ajustaremos el parámetro del número de árboles que se entrenarán con la misma muestra que entrenamos al árbol anterior y con los parámetros óptimos obtenidos para el mismo. La única métrica que utilizaremos para decidir este parámetro será la tasa de acierto.

- Por último, generaremos una nueva muestra de un tamaño mayor con la que entrenaremos el modelo final, que finalmente generará los mapas de saliencia y evaluaremos su capacidad empleando medidas sAUC y sNSS.

Por último, como ya hemos comentado en la introducción, debemos confiar en que el tiempo de respuesta del algoritmo sea lo más corto posible ya que, si queremos aplicar nuestro modelo a procesos reales, estos requerirán de procesos lo más rápidos posibles de tal forma que permitan actuar con eficacia y prontitud ante posibles situaciones de anomalías.

5. Experimentos y resultados

5.1 *Set-up*

El proceso de *set-up* que llevaremos a cabo en nuestros experimentos se distribuye en tres etapas que nos permitirán obtener el modelo objetivo que queremos optimizar. Estas tres etapas son:

1. Optimización de *features*: Realizaremos un submuestreo temporal de los *frames* de los videos y obtendremos sus mapas de características para, posteriormente, optimizarlos en función de algún parámetro. Evaluaremos la selección de dichos parámetros en función de sus resultados para las métricas sNSS y sAUC.

2. Optimización del clasificador: En esta segunda etapa, optimizaremos los parámetros del árbol de clasificación que emplearemos haciendo uso para ello de comparaciones en función de su tasa de acierto o *Accuracy* sobre los datos de validación.

3. Evaluación del sistema: en último lugar, llevaremos a cabo la evaluación del sistema 5-*fold* a través de las métricas utilizadas en la primera etapa.

En los siguientes apartados se describirán detalladamente los pasos que se han llevado a cabo en la realización de cada etapa.

5.2 Optimización de descriptores

En esta sección se analizará la optimización de los descriptores utilizados a través de la selección de sus parámetros. En primer lugar, como se comenta en la sección de modelos *top-down* debemos realizar un análisis previo de las posibles características de los videos que llaman la atención (*fixations*) de los sujetos del estudio preliminar, para ello, creamos una lista de las que, pensamos, serán más generales y, por lo tanto, ayudarán a crear el modelo.

5.2.1. Descriptor de magnitud

Antes de tener completa la información de este descriptor, necesitamos suavizar las imágenes de tal forma que las zonas espúreas queden filtradas y se obtenga un mapa suave más real y ajustado al modelo.

Para este propósito, emplearemos un filtrado de tipo Gaussiano empleando un filtro al que tendremos que pasar como parámetros el propio *frame* de magnitud y un parámetro sigma que represente el valor de la desviación del filtro paso bajo a emplear. Para optimizar dicho parámetro, estudiamos diferentes valores del mismo utilizando las métricas sAUC y sNSS analizadas, obteniendo los siguientes resultados mostrados en las figuras 14 y 15:

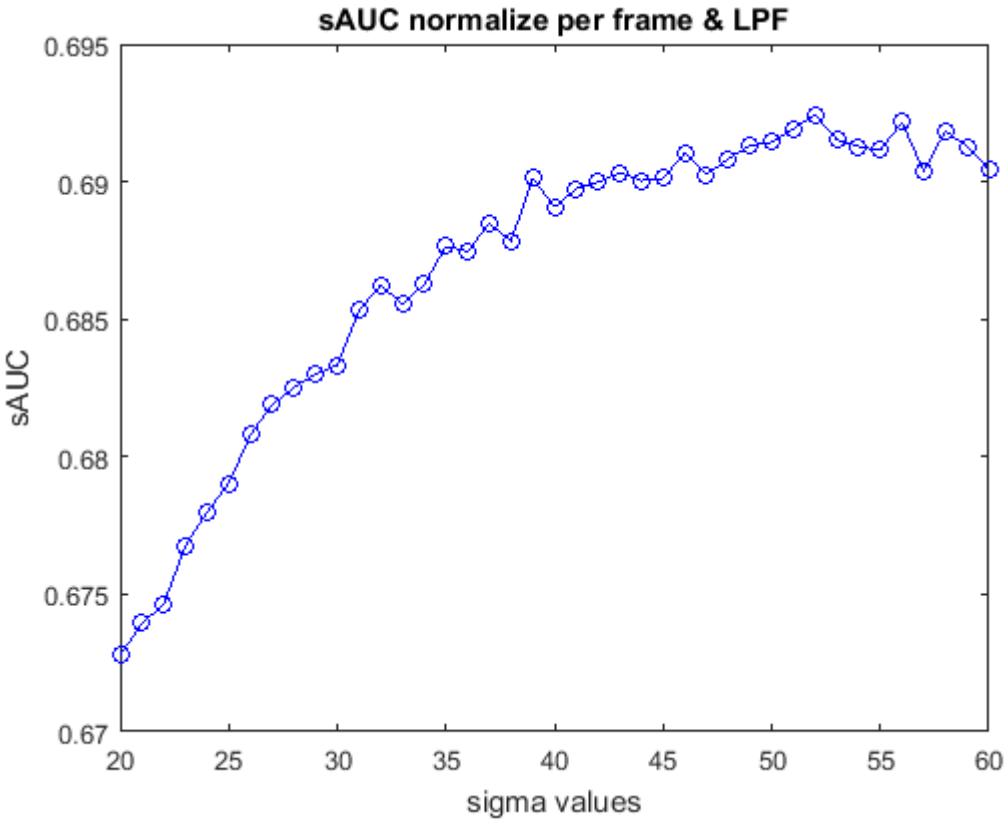


Figura 14. Valores de sAUC según valor del parámetro sigma

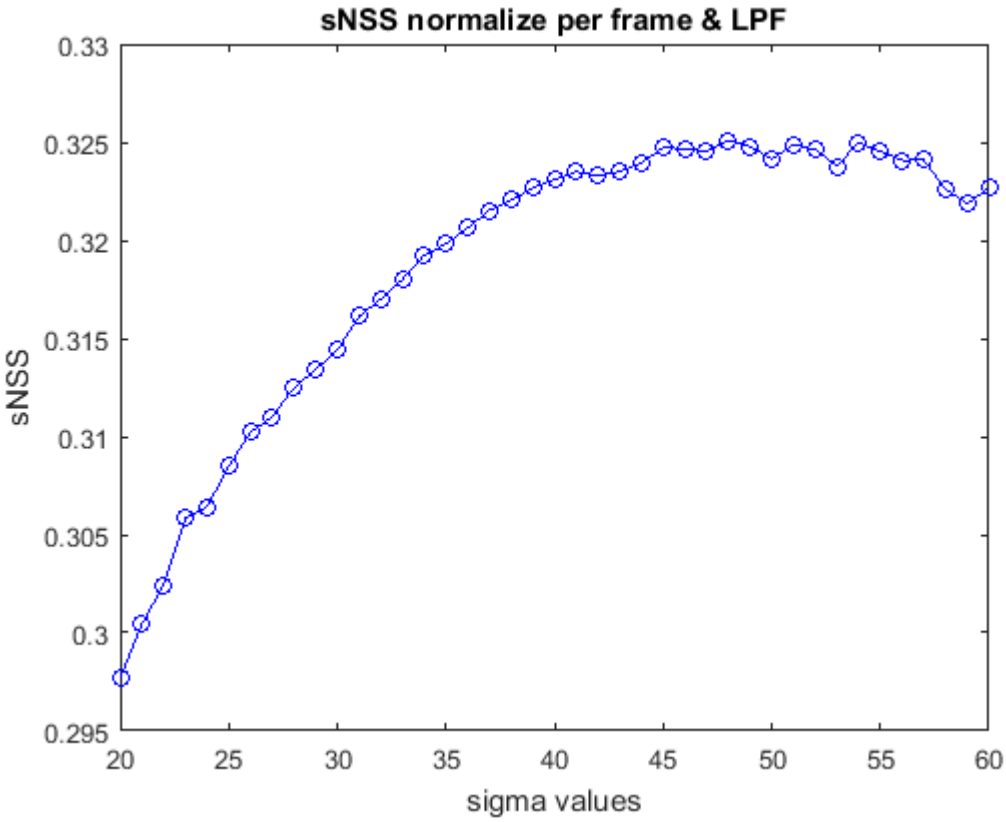


Figura 15. Valores de sNSS según valor del parámetro sigma

Estando situado el máximo valor de sAUC en el valor 52 y el máximo de sNSS en 48, por lo tanto, tendremos dos candidatos a ser el parámetro sigma y tendremos que seleccionarlo dando prioridad a una de las dos métricas, esta comparación se observa en la tabla 2.

Valor de sigma	**s-AUC**	**s-NSS**
52	0.6924	0.3246
48	0.6908	0.3251

Tabla 2. Comparación de resultados para los valores candidatos

Finalmente, seleccionamos como valor para el parámetro sigma 52, ya que el valor de sNSS es menos inferior respecto al valor de sNSS del valor 48 que el valor de sAUC para el valor 48 respecto al de 52. Con esta sigma y aplicando el filtrado paso bajo, obtendríamos un mapa para el *frame* de estudio como el de la figura 16, en la que se muestran, por este orden: *frame* original con los vectores de movimiento, mapa generado sin procesar y mapa de atención final:

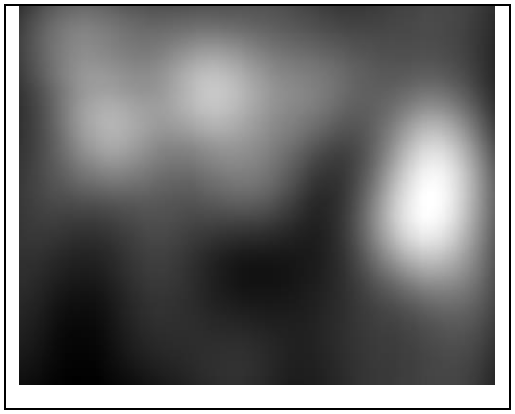


Figura 16. Mapa de saliencia para la característica de magnitud

5.2.2. Descriptor de aceleración

De la misma forma que en el caso de la magnitud, el mapa obtenido tendrá zonas espúreas de movimiento que debemos eliminar utilizando, de nuevo, un filtrado paso bajo gaussiano cuya desviación, debemos optimizar. Además, podemos observar que la información que este nos aporta discrimina unos movimientos de otros, ya que dará más valor a las zonas con cambios instantáneos y continuos (ventanas) y al propio suelo (sombras). Para el caso de la aceleración, repetimos de nuevo el mismo proceso utilizado para optimizar la sigma de la *feature* magnitud y obtenemos los resultados para las métricas sAUC y sNSS mostrados en las figuras 17 y 18, respectivamente y recogidos en la tabla 3:

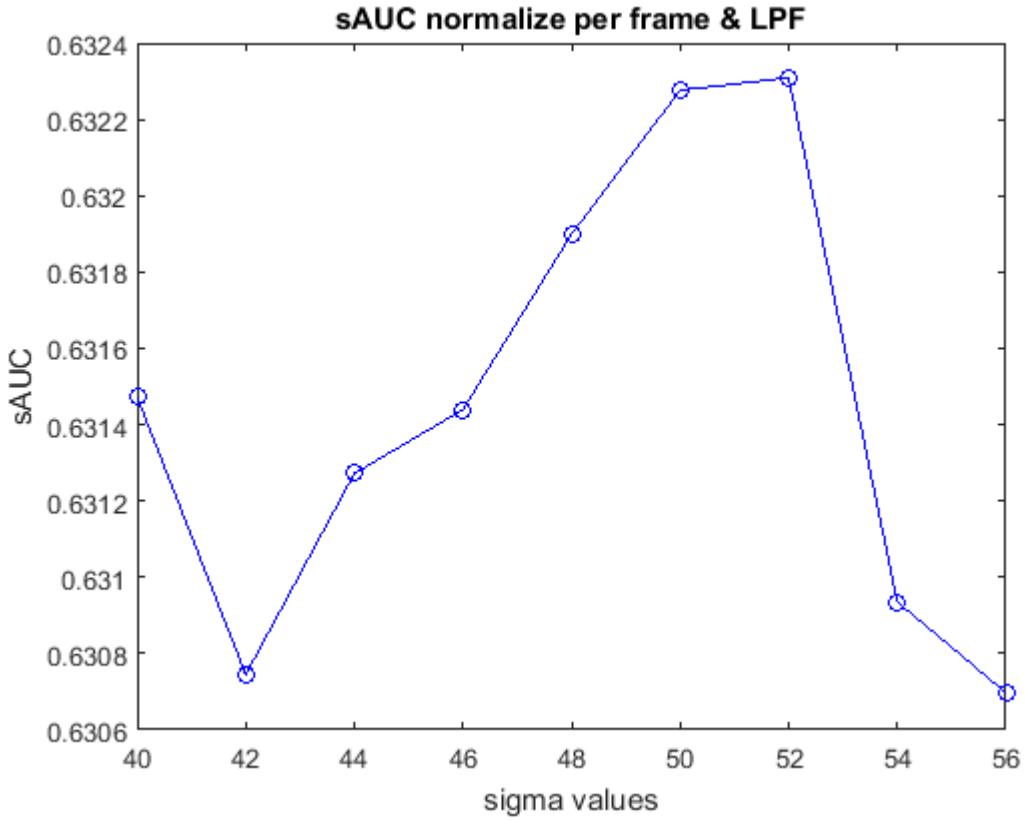


Figura 17. Valores de sAUC según valor del parámetro sigma

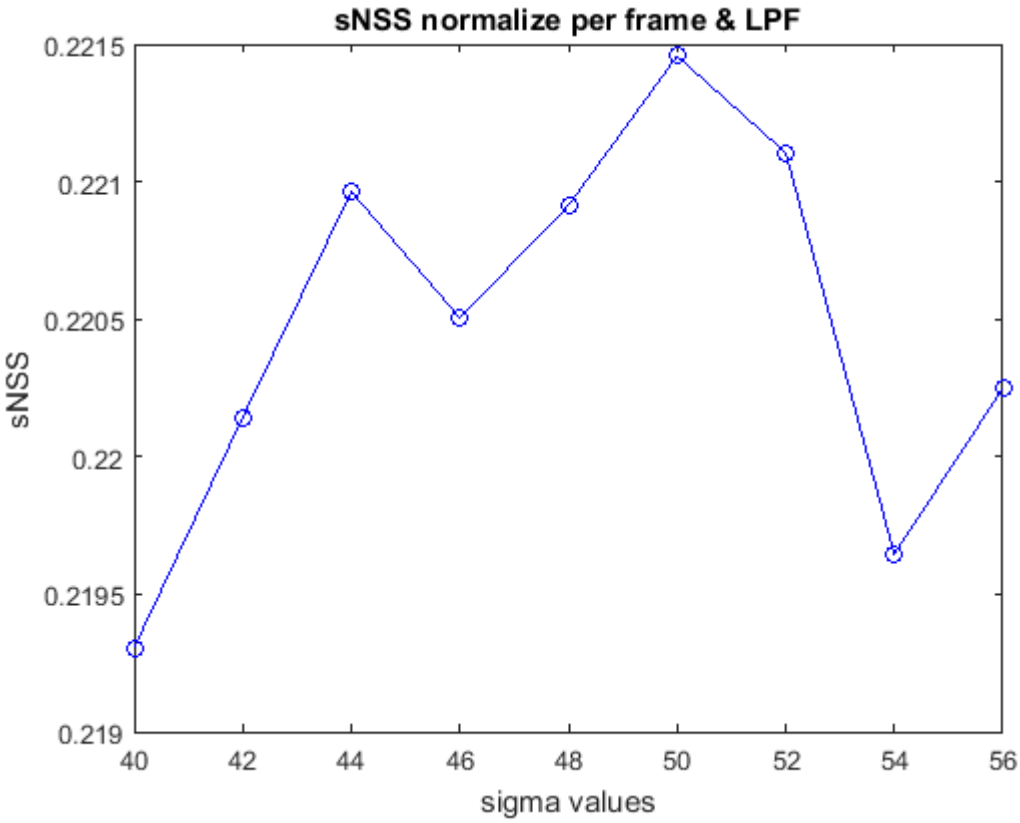


Figura 18. Valores de sNSS según valor del parámetro sigma

Valor de sigma	**s-AUC**	**s-NSS**
52	0.6323	0.2211
50	0,6322	0.2215

Tabla 3. Comparativa de resultados para la selección del parámetro sigma

Una vez más tenemos dos candidatos a ser el valor definitivo de la varianza de nuestro filtro gaussiano y, de nuevo, utilizamos la misma argumentación para seleccionar el definitivo, que será 52. Empleando el filtrado sobre nuestro *frame* de ejemplo, obtendríamos el mapa de la figura 19, en la que se muestran, por este orden: *frame* original con los vectores de movimiento, mapa generado sin procesar y mapa de atención final:

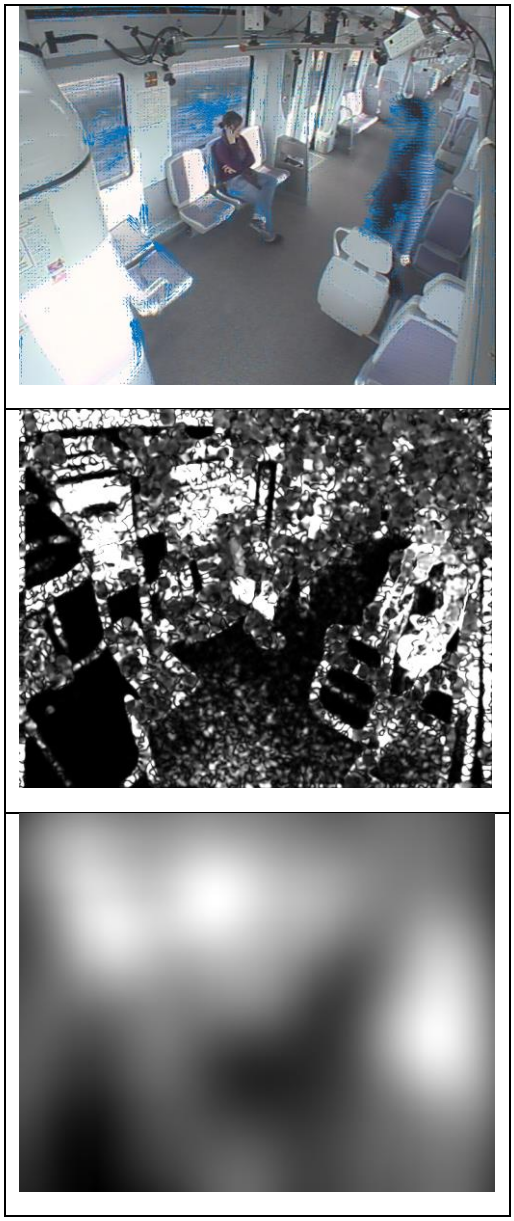


Figura 19. Ejemplo de mapa de saliencia para la feature aceleración

5.2.3. Center-Bias y shuffle map

Para modelar el sesgo de la visión humana que consiste en dirigir la atención hacia el centro de la imagen, utilizaremos una gaussiana bidimensional que, situada, en el centro de la imagen, consiga los mejores resultados para las métricas utilizadas. En primer lugar, vamos a realizar el experimento para una gaussiana simétrica que utilizaremos como mapa de saliencia para todo el conjunto de *frames* y, a continuación, utilizaremos con una gaussiana con distinta desviación en cada eje.

Para nuestro primer caso, en el que la gaussiana es simétrica, obtenemos los resultados mostrados en las figuras 20 y 21:

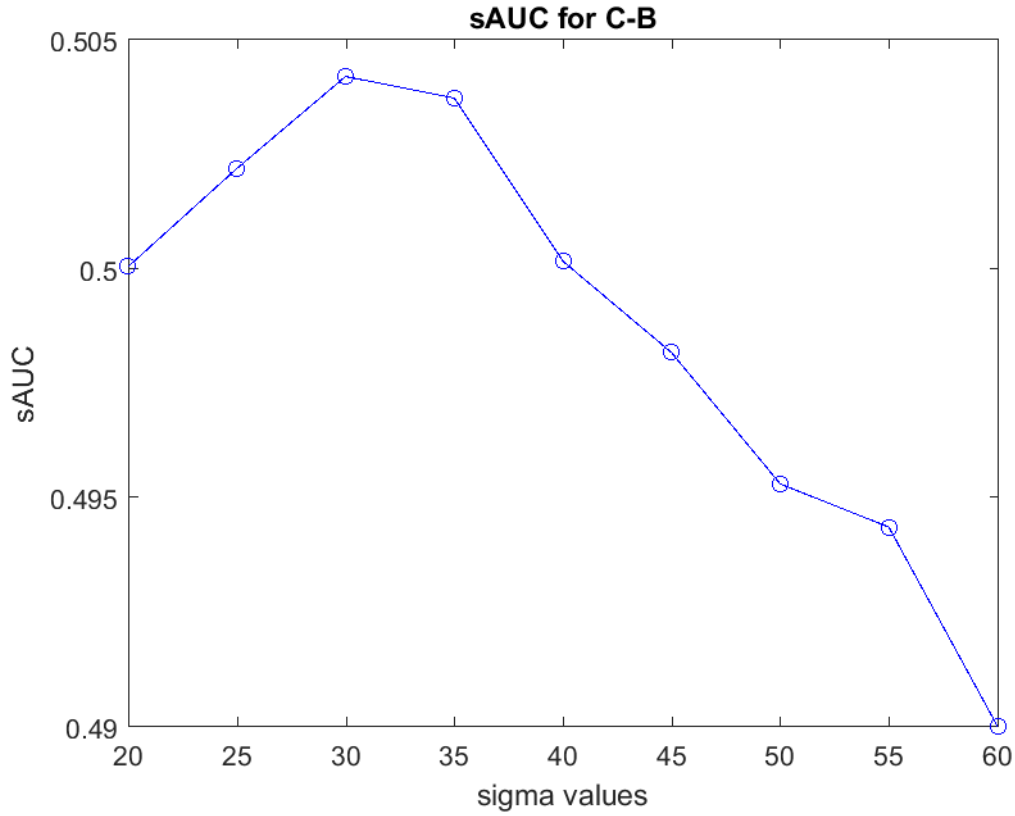


Figura 20. Valores de sAUC según valor del parámetro sigma

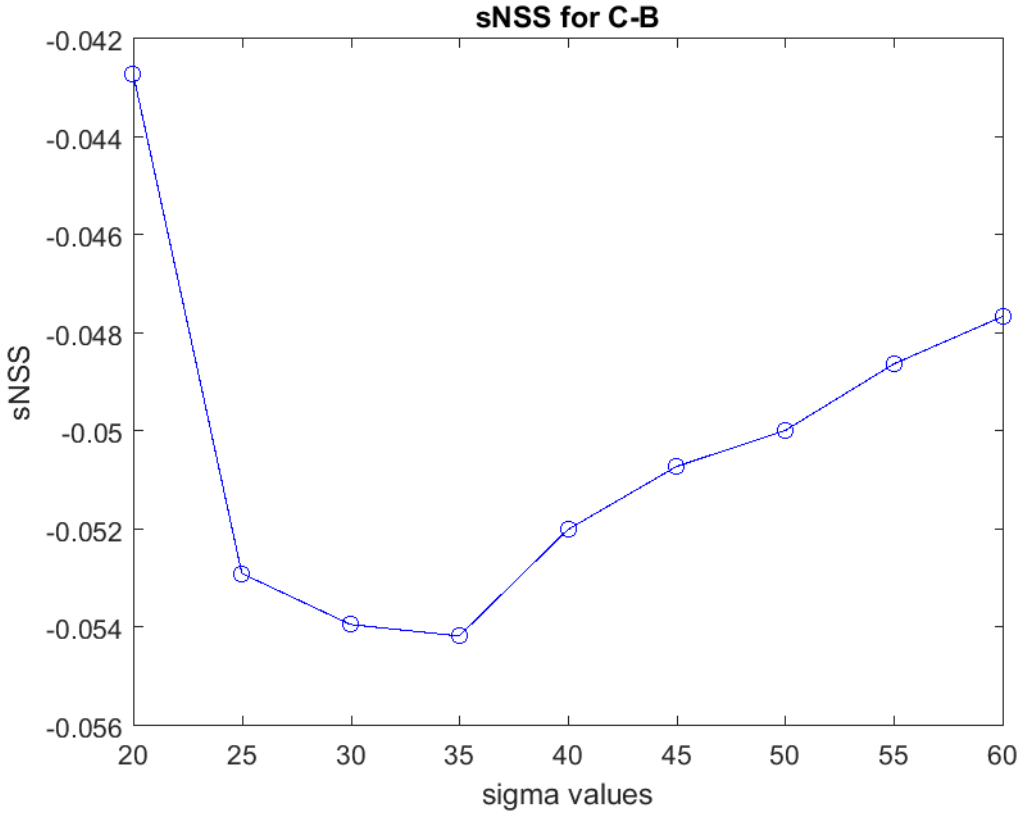


Figura 21. Valores de sNSS según valor del parámetro sigma

Al ser los valores de la métrica sNSS tan bajos, tomaremos como valor candidato el de mayor métrica sAUC, siendo su valor el que se muestra en la tabla 4:

Sigma	**sAUC**	**sNSS**
30	0.5043	-0.0540

Tabla 4. Comparativa de los resultados para Center-bias simétrico

Al realizar el mismo experimento, pero modelizando el *center-bias* con una gaussiana bidimesional no simétrica, obtenemos, para 25 combinaciones diferentes de valores, los siguientes resultados mostrados en las figuras 22 y 23:

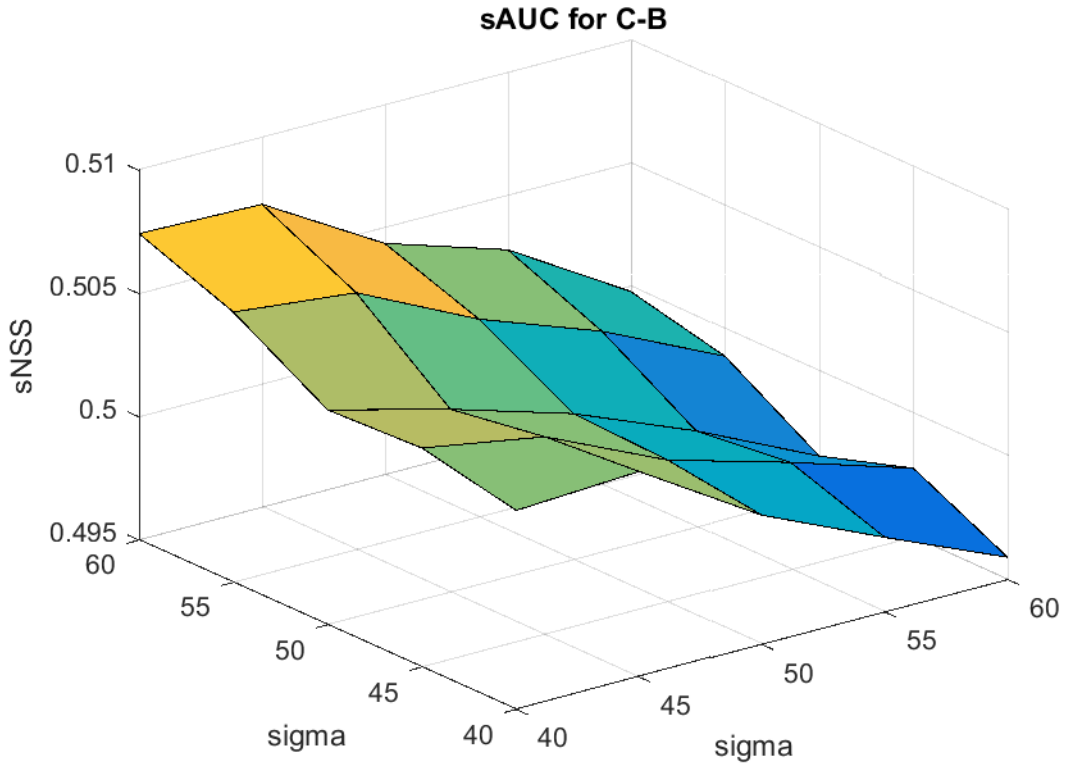


Figura 22. sAUC para las posibles combinaciones de sigma

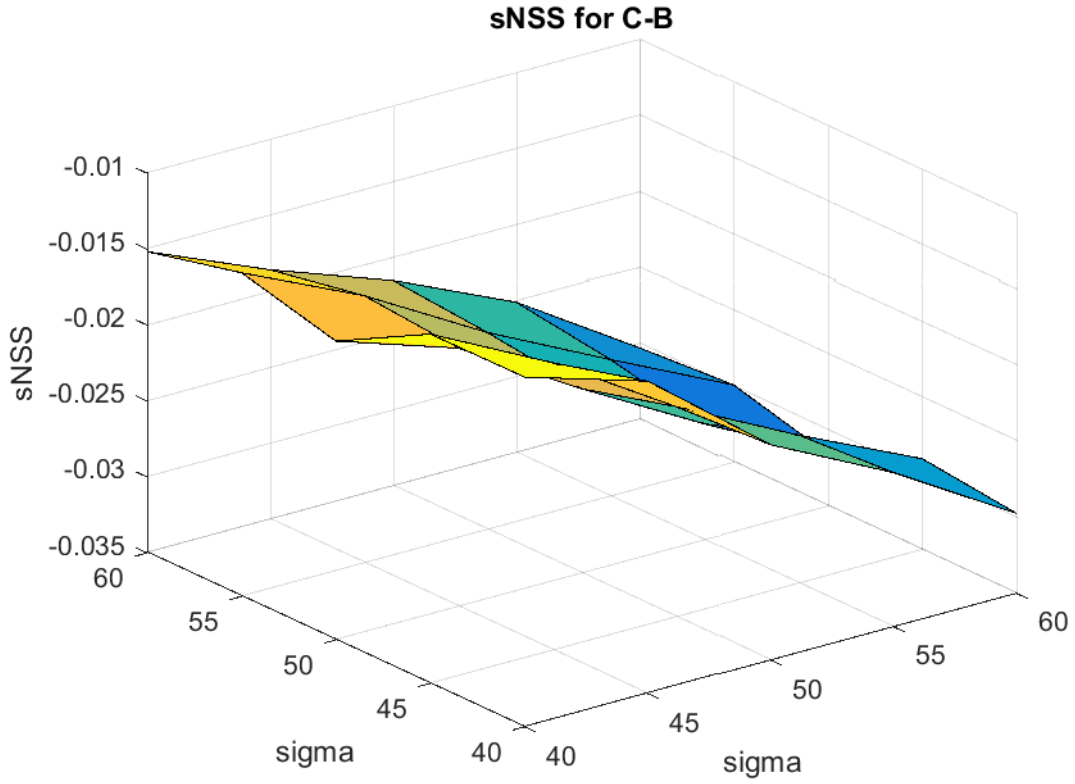


Figura 23. sNSS para diferentes combinaciones de sigma

En este caso, las combinaciones con valores más altos serán la 5 para la métrica sAUC y la 3 para el caso de sNSS. Atendiendo a estas dos combinaciones candidatas, tendríamos la siguiente comparación recogida en la tabla 5:

# Combinación	**Sigmas**		**sAUC**	**sNSS**
5	**X = 40**	**Y = 60**	0,5080	-0,0147
3	**X = 40**	**Y = 50**	0,5042	-0,0148

Tabla 5. Comparativa entre los valores obtenidos para la gaussiana bidimensional

De ambos experimentos podemos sacar la única conclusión de que lo mejor será emplear una gaussiana con una desviación de 40 de ancho y 60 de alto, siendo, aun así, los valores de ésta muy límites. El mapa resultante para modelizar el *center-bias* sería el que se muestra en la figura 24:

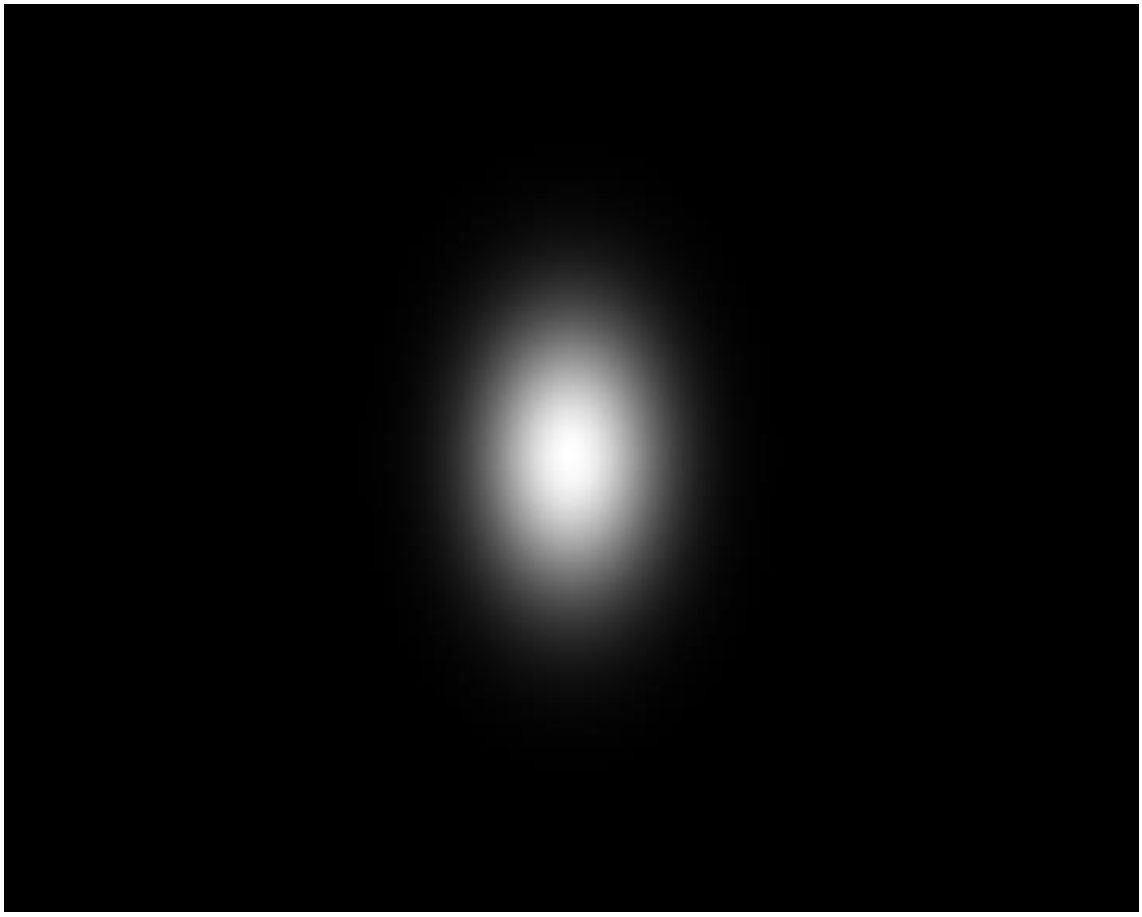


Figura 24. Resultado de nuestro modelado del Center-bias

A la vista de los resultados obtenidos, con un escaso 0.5 en métrica sAUC y un -0.015 de sNSS, el *center-bias* no es una característica relevante para nuestro problema y, por lo tanto, debemos intentar modelar el sesgo de la atención con otro mapa que resulte más apropiado en nuestro caso. Para ello, vamos a realizar un experimento con un único mapa de saliencia que será nuestro propio shuffle map, con él, intentaremos explicar las regiones de saliencia como probabilidades (frecuentistas) de fijaciones del ojo humano en este experimento. Los resultados obtenidos son los que se encuentran en la tabla 6:

SHUFFLE MAP	
sAUC	0.5460
sNSS	0.0866

Tabla 6. Resultados obtenidos de emplear el shuffle map como característica

A la vista de estos resultados, utilizaremos el shuffle map como una *feature* común a todos los *frames* para obtener un mapa de saliencia, con el único inconveniente de que este, queda limitado al contexto de nuestro problema, aunque, este podría usarse en otros videos ya que modeliza un aspecto de la visión humana, al fin y al cabo.

5.2.4. Detección de pasajeros

Para realizar la optimización de este descriptor, teníamos que superar dos experimentos relacionados, en primer lugar, debemos encontrar el detector que mejor se ajuste a nuestro

caso y nos proporcione garantías y, en segundo lugar, debemos optimizar los parámetros propios de este.

Utilizando un detector de personas clásico, podemos detectar pasajeros en forma de coordenadas que forman los *bounding box*, es decir, el detector devuelve los puntos en los que es más probable que exista un pasajero y, además, sigue a los pasajeros mientras continúe detectándolos en el video. Este algoritmo es interesante y resultaría el elegido para nuestro propósito excepto porque únicamente detecta pasajeros en posición vertical con la cámara estática. Si comparamos la naturaleza de ambas detecciones encontramos con que este algoritmo es óptimo para imágenes como la que se muestra en la figura 25:

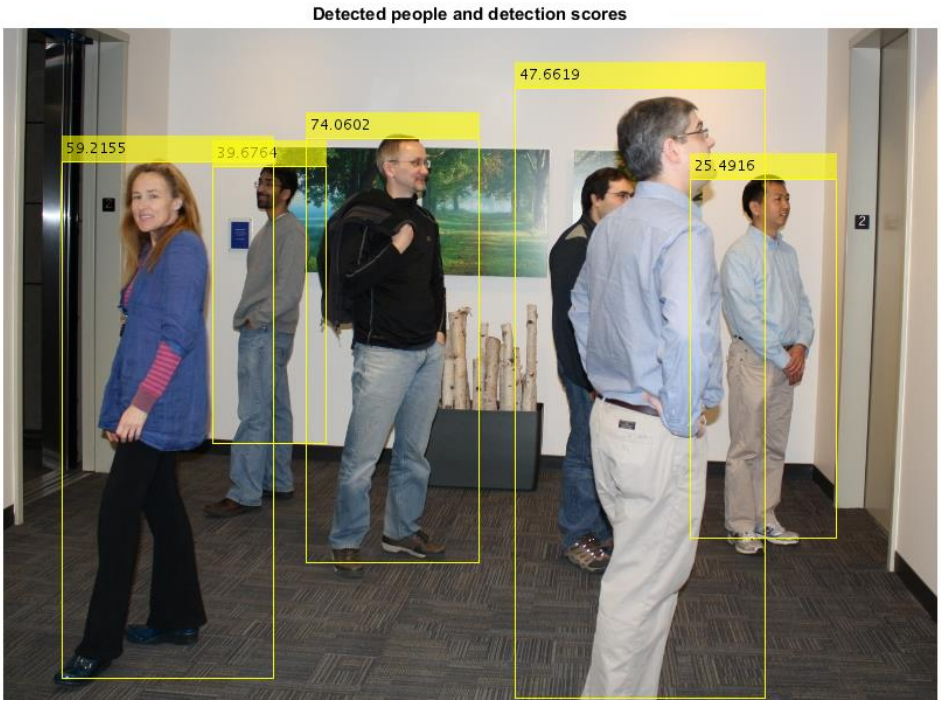


Figura 25. Ejemplo de detección con pasajeros en vertical

En cambio, en videos como el nuestro, la detección es mala, ya que los pasajeros no siempre se encuentran estrictamente en posición vertical y, además, no son imágenes tan nítidas y de buena resolución como para captar pasajeros a través de unas determinadas característica, podemos observar un ejemplo del funcionamiento del mismo en la figura 26:

Figura 26. Ejemplo de detección errónea

En la figura 26 se puede observar las limitaciones del algoritmo, en la que detecta, en vertical, una zona en la que no hay ningún pasajero, pero si movimiento.

Esto quiere decir que tendremos que abarcar el problema de otra forma ya que ni el detector de caras ni el de pasajeros nos darán buenos resultados. Una forma de hacerlo es realizar una detección de objetos basada en movimiento (*motion-based*) que, de alguna forma detectará aquellas zonas en las que exista un movimiento y pueda estar provocada por un objeto o persona y nos proporciona un seguimiento del mismo que será clave para la resolución de nuestro problema. Como ya se ha explicado en el apartado 4.1.3., encontramos un buen detector que emplea mezcla de gaussianas y filtros de Kalman para este propósito.

Este algoritmo nos proporciona, en resumen, una lista de *bounding boxes* en cada *frame* y numerosas características de los mismos, como sus coordenadas (alto y ancho) y su duración, número de *frames* en los que se mantiene dicho objeto. Además, permite a los objetos "desaparecer" durante un número limitado de *frames* sin perder su detección, es decir, una persona puede salir del plano de la imagen y volver a entrar durante un periodo de tiempo y su detección seguirá asociada a la primera detección, al primer *bounding box*. Tener esta información nos va a permitir discriminar mejor la detección puesto que, para nosotros, resultan irrelevantes muchos movimientos que suceden en el tren, como puede ser el movimiento en las ventanas o el de las sombras en el suelo y por ello, seleccionaremos los *bounding boxes* que se mantengan durante un número mínimo de *frames*. Para elegir este parámetro de manera adecuada, debemos analizar el parámetro *age* de los *bounding boxes* y generar mapas de saliencia en función de los mismos para después evaluar sus métricas sAUC y sNSS. Un ejemplo

de los mapas que se generaran es el siguiente, en él, centramos en cada *bounding box* una gaussiana con una desviación fija de 52. Un ejemplo de estos mapas sería el mostrado en la figura 27:

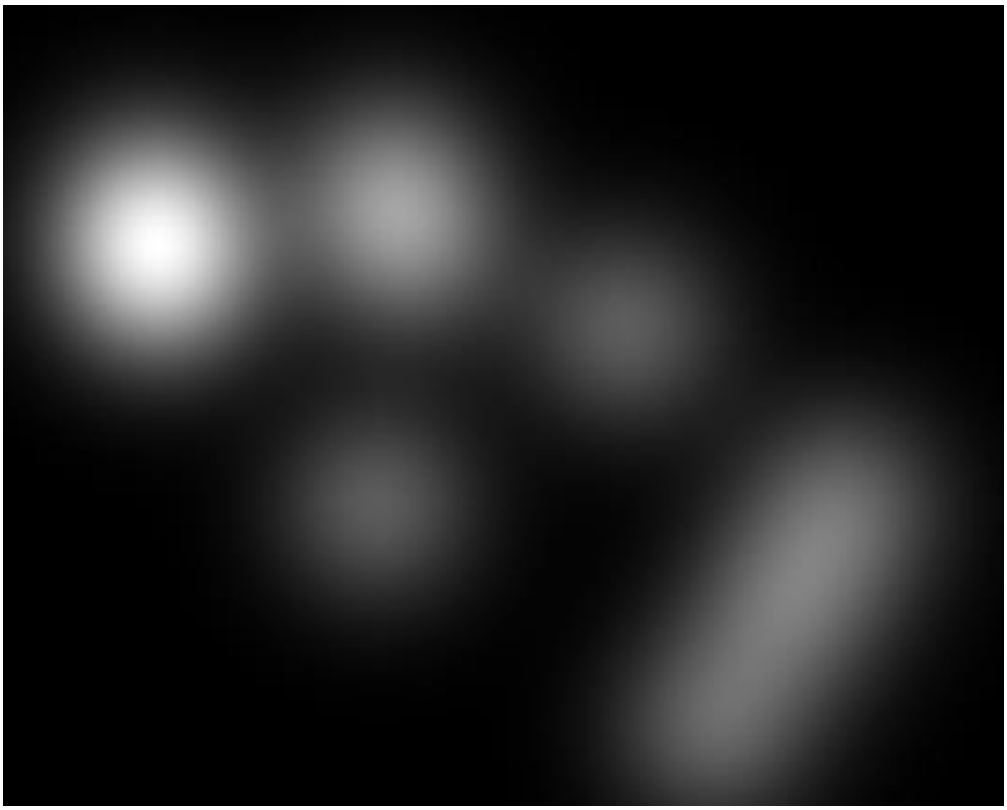


Figura 27. Ejemplo de mapa basado en detección de pasajeros, superponiendo gaussianas en el centro.

A la izquierda tenemos los centros de cada *bounding box* presentes en el plano y a la derecha le superponemos las gaussianas para crear el mapa de saliencia. Los resultados obtenidos por lo tanto para elegir el parámetro *age* son los mostrados en las figuras 28 y 29:

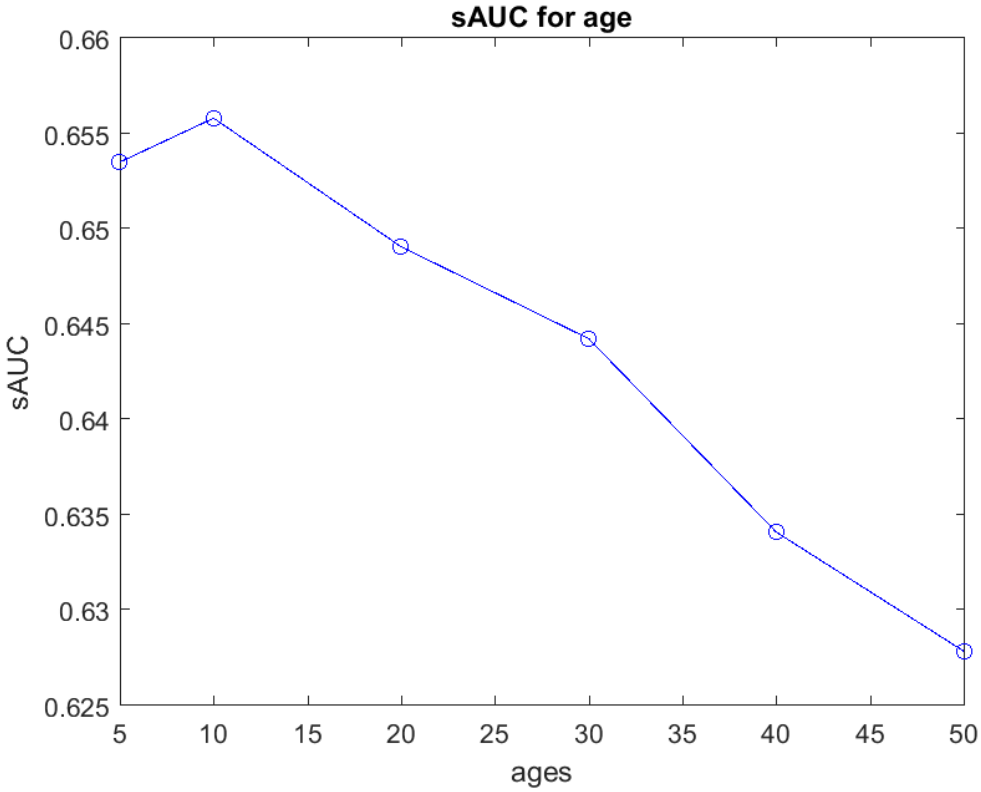


Figura 28. Valor de sAUC según parámetro age

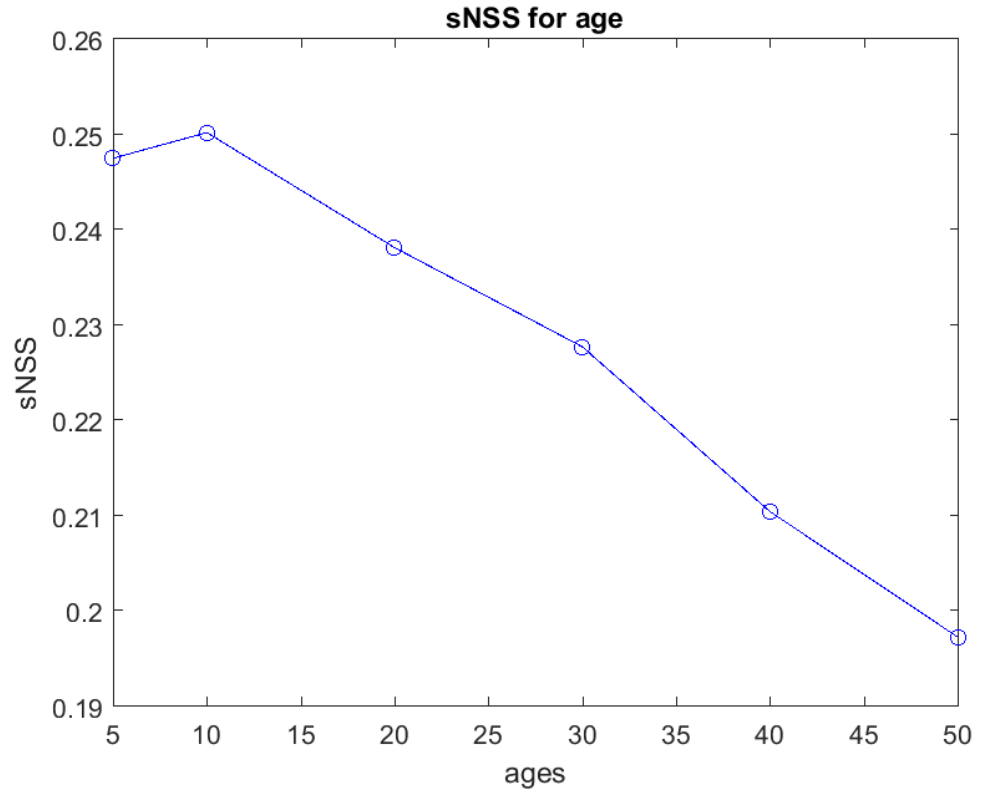


Figura 29. Valor de sNSS según parámetro age

En este caso, evidentemente, el valor de *age* que debemos seleccionar es 10, que nos ofrece unos resultados recogidos en la tabla 7:

Umbral de age	sAUC	sNSS
10	0,6558	0,2502

Tabla 7. Resultado final de parámetro age

Una vez que el parámetro *age* ha sido fijado, pasamos a modelar el contenido que situaremos dentro de ese *bounding box* para generar el mapa de saliencia. Como hemos dicho y porque será lo más preciso, situaremos una gaussiana centrada en el propio *bounding box* y trataremos de optimizar su desviación a lo largo de ambas direcciones x e y. En primer lugar, se valorará el resultado para un caso simétrico en el que la gaussiana tomará valores entre 20 y 60. Los resultados de este primer experimento son los mostrados en las figuras 30 y 31:

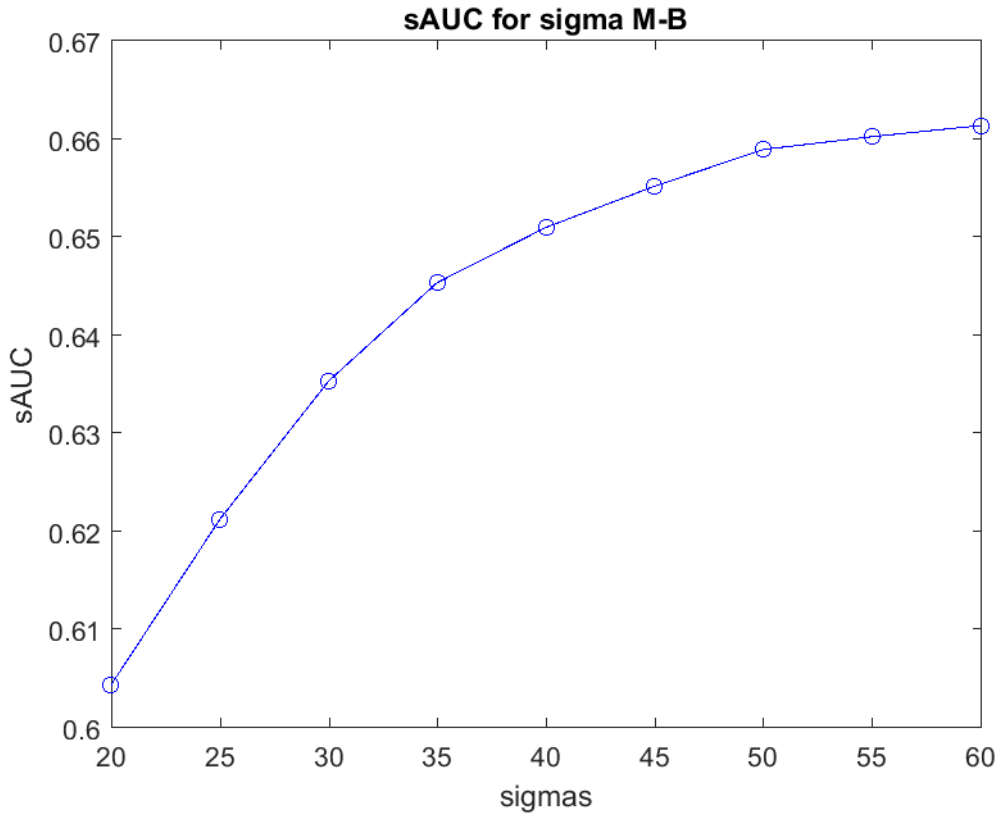


Figura 30. Valor de sAUC según parámetro sigma

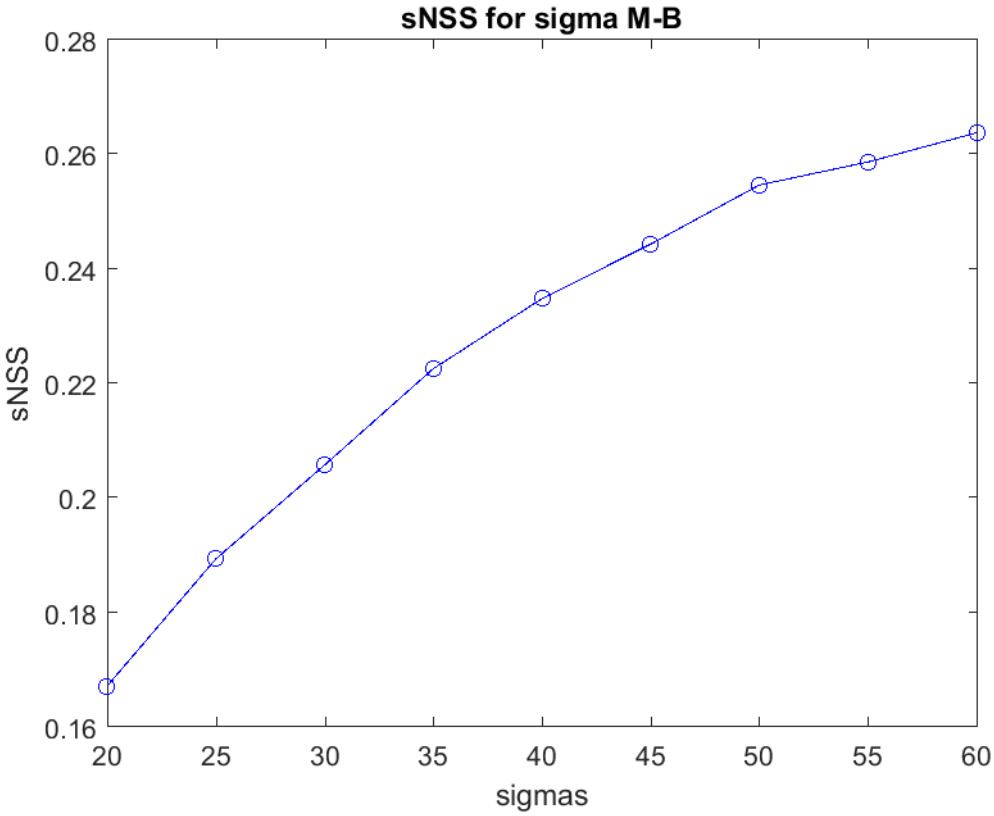


Figura 31. Valor de sNSS según parámetro sigma

Siendo nuestro valor candidato el mayor, es decir, 60, con el que hemos obtenido los resultados de la tabla 8:

Sigma	sAUC	sNSS
60	0,6614	0,2638

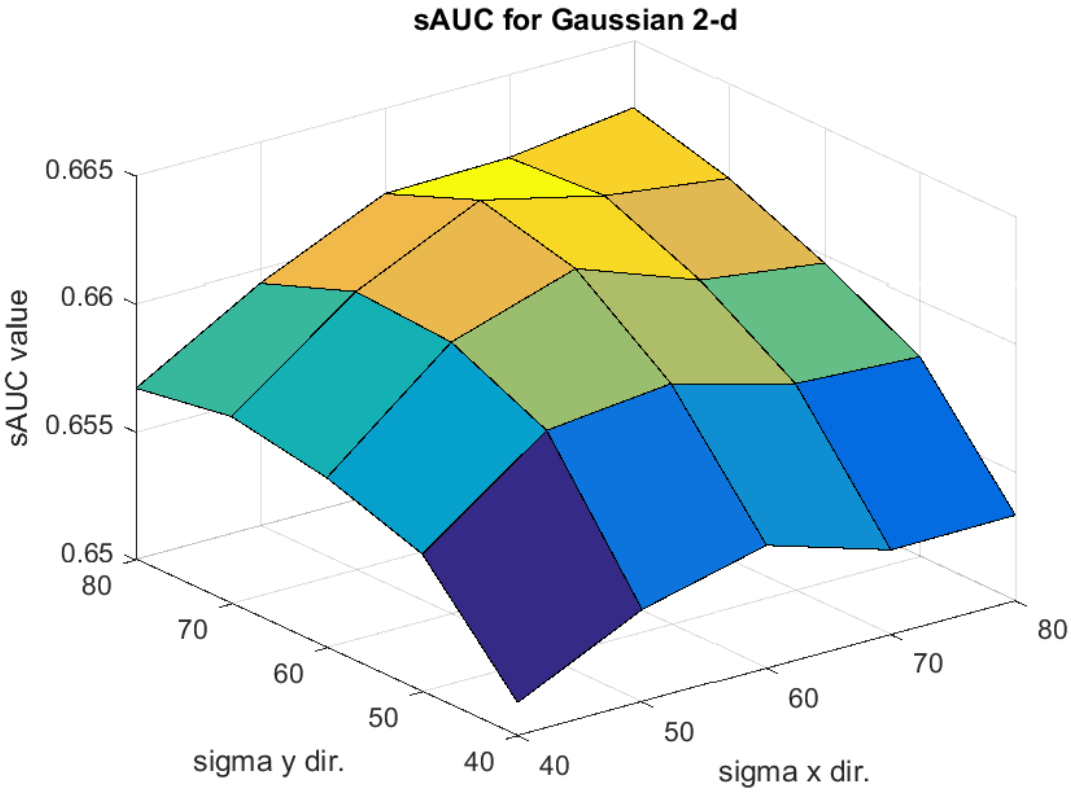
Tabla 8. Resultados de la elección del parámetro sigma

En segundo lugar, para seguir ajustando nuestra gaussiana al *bounding box* correspondiente, trataremos diferentes combinaciones entre posibles valores de desviaciones, de tal forma que la región que estamos modelando se aproxime más a un pasajero, es decir, en video, si un *bounding box* es más largo que ancho, posiblemente se trate de un pasajero de pie y por lo tanto la gausiana tendría más sentido con mayor desviación en el eje vertical que en el horizontal. Los resultados de este experimento son los siguientes:

Por último, para cerrar la *feature* de la detección basada en movimiento, haremos un último experimento que consistirá en modelar la gaussiana utilizando como desviación en el eje vertical el alto del *bounding box* y como desviación en el eje horizontal el ancho del mismo. Así, y

Figura 32. Valor de sAUC para combinaciones de sigmas

utilizando el mismo procesado que en los casos anteriores, obtenemos los siguientes resultados, que se muestran en las figuras 32 y 33:



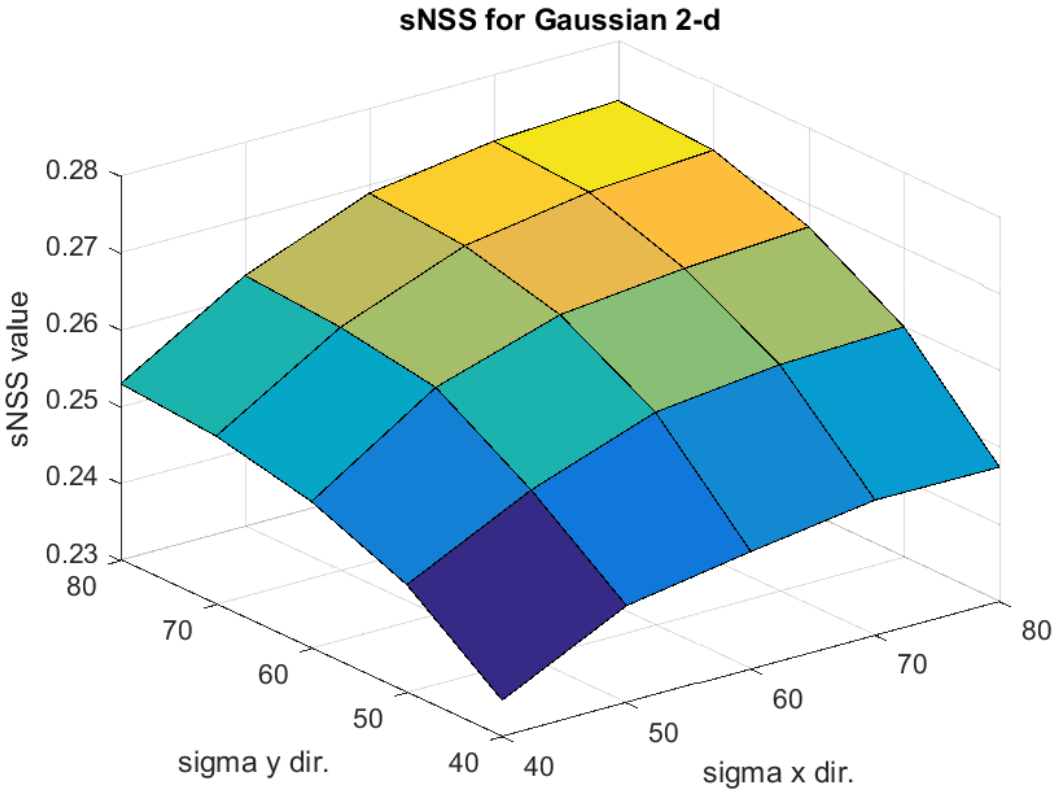


Figura 33. Valor de sNSS para distintas combinaciones de sigma

En este caso, la combinación que resulta candidata es la número 13 que, coincide con la desviación obtenida en el primer apartado ya que trata de una gaussiana simétrica con sigma igual a 60. Los valores de sAUC y sNSS son los de la tabla 9:

Sigma	sAUC	sNSS
60	0,6622	0,2629

Tabla 9. Resultados de selección del parámetro sigma en la gaussiana bidimensional

Tenemos ya una conclusión para nuestro problema sobre la gaussiana y es que, si generalizamos, nuestro *bounding box* deberá modelarse a través de una gaussiana de sigma igual a sesenta, pero, gracias al algoritmo utilizado, tenemos más información acerca de cada *bbox* que podremos utilizar en nuestro propósito. A continuación, vamos a realizar un último experimento que ajuste la gaussiana al *bounding box* correspondiente dando el valor a la desviación del eje X a partir del ancho del *bounding box* y, lo mismo para el eje Y y el alto del mismo. Los resultados obtenidos por video son los mostrados en las figuras 34 y 35:

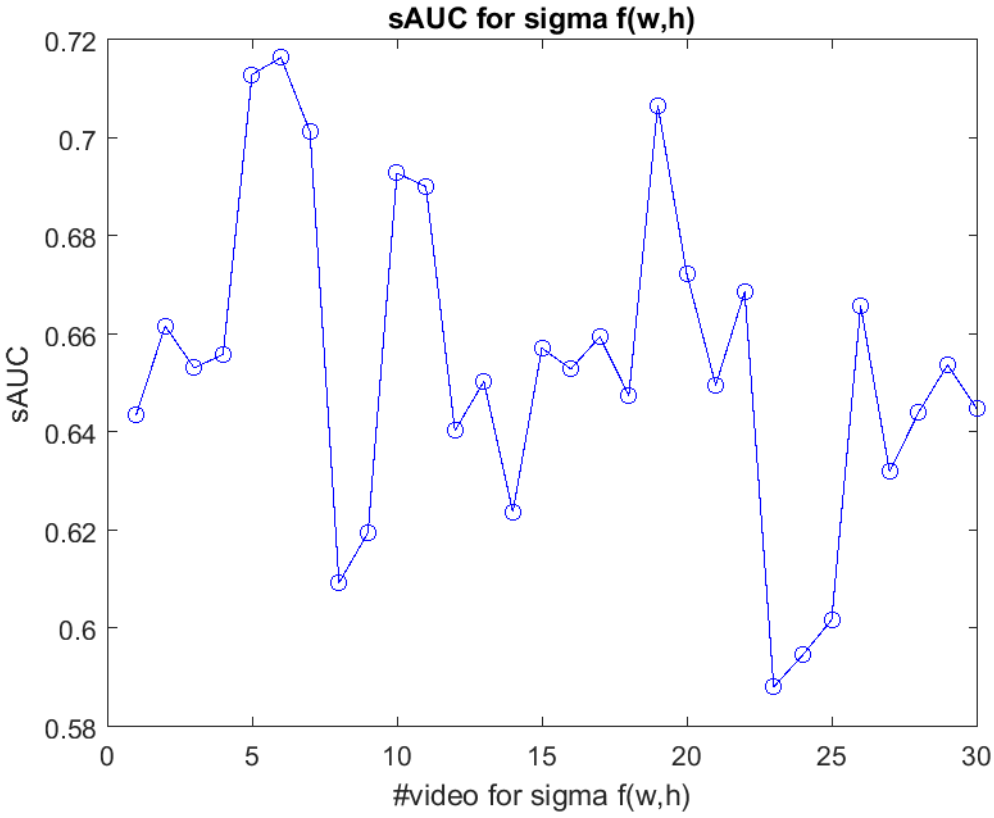


Figura 34. valores de sAUC para sigmas en función del alto y el ancho

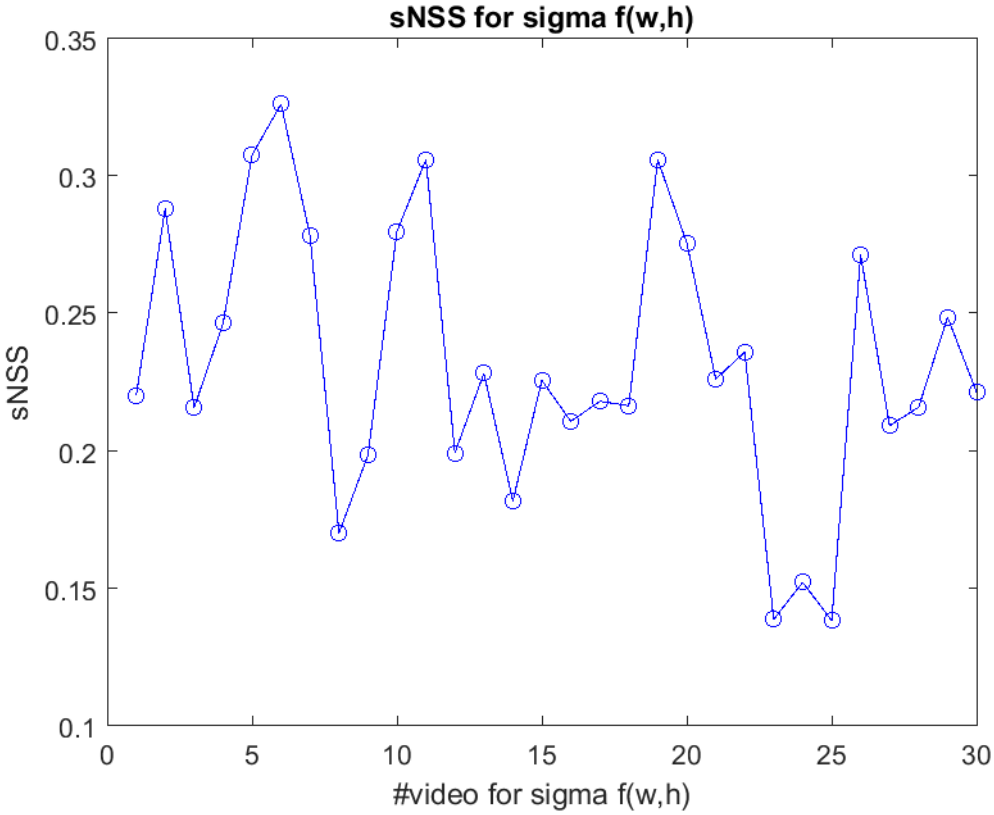


Figura 35. Valor de sNSS para sigmas en función del alto y el ancho

Como vemos, de este resultado tendría sentido hablar del valor medio obtenido por video para poder compararlo con los anteriores y así decidir cuál sería la mejor gaussiana a utilizar. Estos valores medios se encuentran en la tabla 10:

Sigma	**sAUC**	**sNSS**
F(width,height)	0.6537	0.2320

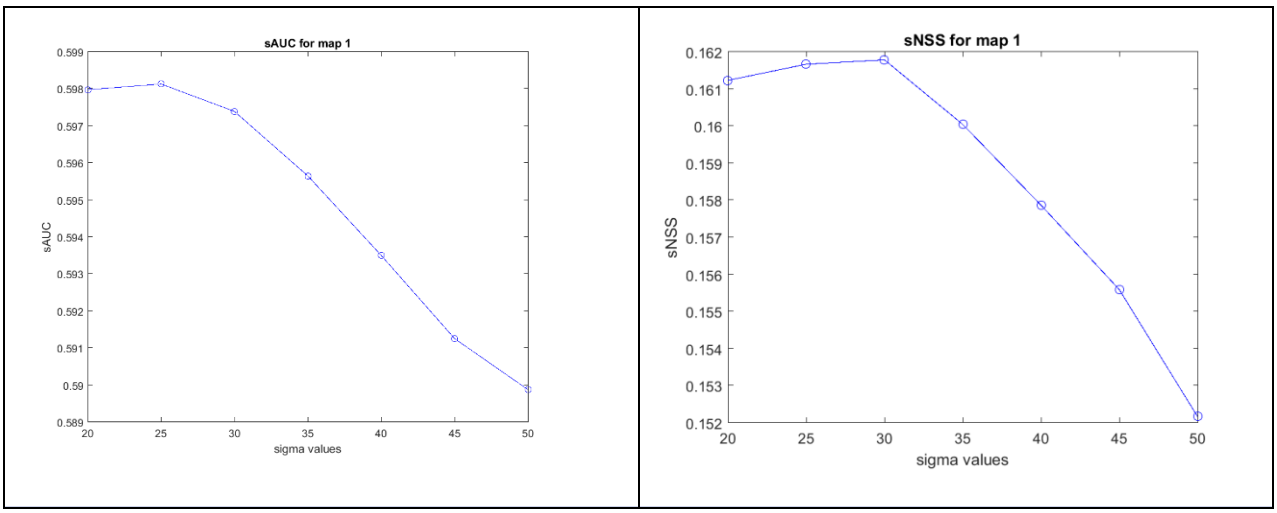
Tabla 10.. Resultados para gaussianas modeladas según ancho y alto

Finalmente, tras las pruebas realizadas para la característica que llamaremos *motion based,* seleccionaremos una gaussiana simétrica cuya desviación sea de valor 60 y, solo tomaremos como detecciones válidas aquellas que estén durante un número igual o mayor de diez *frames* activas.

5.2.5. Características bottom-up del modelo GBVS

A partir de los mapas de características de tipo *bottom*-up, podemos llevar a cabo una serie de optimizaciones para mejorar los futuros resultados. De ellos se puede ver como la resolución es notablemente inferior, de apenas 32 x 26 pixeles. Esta rareza constituye además una nueva prueba a la que debemos someter a los mapas para fijar sus características más propicias ya que será necesario realizar las pruebas con y sin reescalado para tener el mismo tamaño que el *frame* original. Los diferentes experimentes que se han llevado a cabo son los siguientes:

1. Reescalado de cada mapa y suavizado. Independiente:



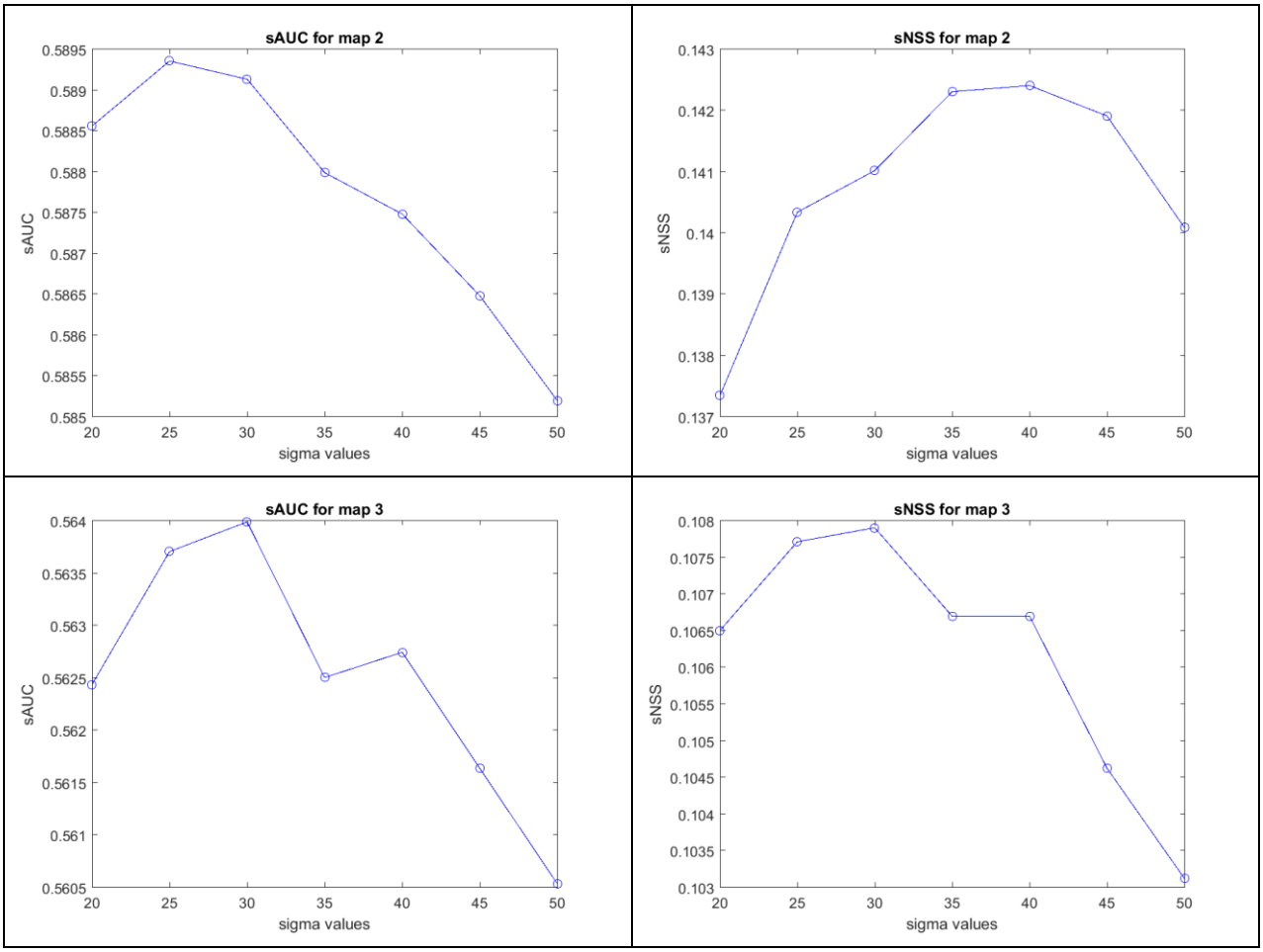


Figura 36. Valores de sAUC y sNSS para cada mapa con este proceso

A la vista de la figura 36, el valor de sigma que resultaría más apropiado, pues coincide en mapa 1 y mapa 2, que son los resultados más altos, es 25. Las métricas resultantes para el valor 25 serían los que muestra la tabla 11:

	Mapa 1	**Mapa 2**	**Mapa 3**
sAUC	0,5890	0,5806	0,5597
sNSS	0,1403	0,1120	0,1017

Tabla 11. Resultados del primer procesado para cada mapa

2. No reescalado y no suavizado.

En este caso, al no realizar un reescalado previo, filtrar el mapa resulta absurdo debido a su escasa resolución (32x26) y, por ende, no hay que realizar ninguna optimización de parámetros en su sentido. Sin embargo, resulta útil conocer las métricas resultantes ya que, al no reescalar, estaríamos ahorrando tiempo de procesamiento. Con esto, los resultados obtenidos son los de la tabla 12:

	Mapa 1	Mapa 2	Mapa 3
sAUC	0,5898	0,5826	0,5594
sNSS	0,1464	0,1202	0,1022

Tabla 12. Resultados para el segundo procesado en cada mapa

Podemos observar que los resultados son similares a los obtenidos en el caso previo, esta cuestión se debe a que la propia métrica utilizada implementa un reescalado de los mapas para que tengan la misma resolución que el plano de las fijaciones, por lo tanto, resulta evidente que será mejor utilizar el segundo caso ya que ganaremos en tiempo computacional. Por otra parte, el valor resultante de las métricas es el deseado, pues se trata de unos mapas de características *bottom-up* y los resultados están cerca de ser los esperados.

5.3. Optimización del árbol de clasificación

Una vez que la optimización de las características ha concluido, podemos dar paso al clasificador. Como ya se ha descrito, utilizaremos un árbol de clasificación binaria y, posteriormente, un *random forest*, que no es más que una extensión del primero.

El primer paso es seleccionar la muestra y obtener una tabla con las diferentes evaluaciones de las características y las etiquetas o *labels*. Una vez tenemos los datos a utilizar, dividiremos nuestro conjunto en datos para entrenar el modelo y datos para testear. Utilizaremos una división del 80 % para entrenamiento y el 20% restante para test de un total de 19000 puntos muestreados siguiendo el proceso de selección de muestras descrito anteriormente.

Para conocer la naturaleza de clasificación entrenamos un primer clasificador que nos permita conocer la importancia de los diferentes parámetros y la tasa de acierto que aproximará nuestro modelo. Para ello, realizamos una optimización automática que estará basada en la optimización del error de clasificación mediante variaciones del parámetro que representa el número de hojas mínimo del clasificador. Nuestro entrenamiento consiste en 30 iteraciones con diferentes valores de dicho parámetro y obtenemos un valor óptimo de 60 para una estimación de error objetivo de 0.3196. Estos resultados, se pueden observar en la figura 37:

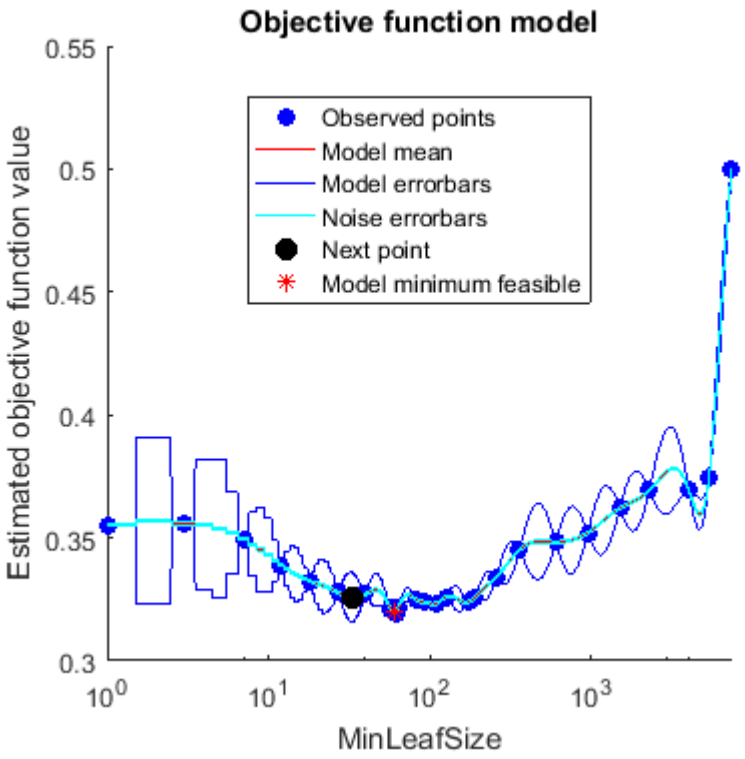


Figura 37. Función objetivo a minimizar

Finalmente, la tasa de acierto obtenida con este modelo automático es de 62.95 %.

Una vez realizado esta primera clasificación, vamos a tratar de optimizar el árbol manualmente utilizando para ello los mismos datos de entrenamiento y test y, modificando diferentes parámetros y métricas. Los parámetros del modelo obtenido y sobre los que haremos modificaciones son los que se encuentran en la tabla 12:

NOMBRE PARÁMETRO	VALOR AUTOMÁTICO	OTROS VALORES
Criterio de separación	Índice de diversidad de Gini	entropía
Número mínimo de nodos (padres)	120	>120
Número mínimo de hojas	60	-
Número de decisiones	14997 (n-1)	-
Criterio de poda (¿Parada?)	error	impureza

Tabla 12. Resumen de los parámetros a optimizar

Podemos encontrar una descripción más detallada en el anexo. Nos quedan así, cuatro combinaciones posibles de árboles sobre las que realizar el análisis, teniendo en cuenta que fijaremos los parámetros de número mínimo de hojas y de padres, así como de decisiones. Las cuatro combinaciones posibles las entrenaremos utilizando la técnica *k-fold* con 5 combinaciones de tal forma que, para nuestro conjunto de datos, explotemos más sus posibilidades. Las tasas de acierto obtenidas son las de la tabla 13:

TIPO DE MODELO	TASA DE ACIERTO
Criterio de separación: Gini **Criterio de poda:** Error	67.34 %
Criterio de separación: Gini **Criterio de poda:** Impureza	67.74 %
Criterio de separación: Entropía **Criterio de poda:** Error	67.51 %
Criterio de separación: Entropía **Criterio de poda:** Impureza	67.49 %

Tabla 13. Resultados de las diferentes combinaciones de parámetros

Para las combinaciones realizadas, hemos obtenido resultados similares y, por lo tanto, hemos de repetir el procedimiento para observar si en algún caso se obtienen más diferencias. Lo que haremos a continuación es proceder de tal forma que el entrenamiento se lleve a cabo con una técnica 2-*fold* (en la que se entrenará con menos datos, 50%) y una técnica 10-*fold* (en la que entrenaremos con más datos). Teniendo en cuenta que de la segunda forma estamos sobreajustando más, el resultado de la primera será clave para seleccionar el modelo ganador. Es importante aclarar en este punto, ya que este procedimiento lo repetiremos en apartados posteriores, que el hecho de entrenar con distintos números de folds, se lleva a cabo simplemente para tener otro criterio más de selección. En última instancia, el modelo final será entrenado utilizando 5-folds.

Los resultados obtenidos para esta segunda prueba se muestran en la tabla 14:

TIPO DE MODELO	T.A. 2-fold	T.A. 10-fold
Criterio de separación: Gini **Criterio de poda:** Error	66.66 %	67.42 %
Criterio de separación: Gini **Criterio de poda:** Impureza	66.80 %	67.90 %
Criterio de separación: Entropía **Criterio de poda:** Error	67.49 %	67.56 %
Criterio de separación: Entropía **Criterio de poda:** Impureza	67.43 %	67.53 %

Tabla 14. Resultados para ajustar el número de folds

Seleccionaremos como valor de los parámetros de separación y poda, los criterios de medida de entropía y error, respectivamente. Finalmente, entrenando un modelo con estas características, podemos encontrar otras métricas que serán interesantes para la caracterización de nuestro árbol de clasificación. Para ello, volveremos a entrenarlo con 5 y 10 *folds*, esta vez lo hacemos para observar que nuestro modelo presenta coherencia y los resultados se mantienen adecuados a nuestro objetivo. Los resultados se recogen en la tabla 15:

	5-*fold*	**10-*fold***
Tasa de acierto media sobre datos test	60.22 %	60.20 %
Tasa de acierto sobre los datos no entrenados	67.26 %	68.01 %
Error de clasificación	32.74 %	31.99 %
Margen medio de separación	19.40%	20.28 %

Tabla 15. Resultados ampliados para distintos folds

Otro punto a considerar en nuestro modelo es la importancia que en él tienen los diferentes predictores (*features*) en la solución final, para ello, podemos ver en forma de gráfica en la figura 38 una representación de esta importancia en función de la variación del MSE en función de estos de tal forma que veamos cual es el más importante y el menos debido a su aportación al error.

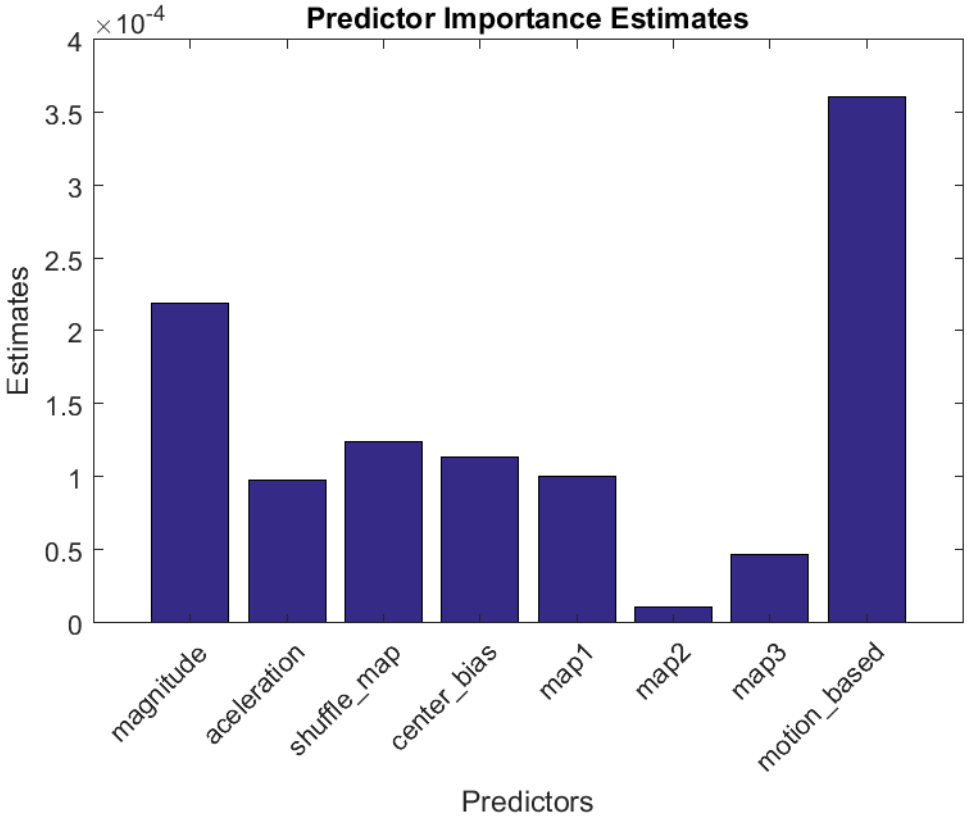


Figura 38. Importancia de cada predictor en el modelo

Como vemos, dos predictores destacan por encima del resto, *motion_based*, en primer lugar y, *magnitude* detrás de él. Además, debemos destacar que los mapas de características *bottom-up* 2 y 3 no nos aportan demasiada importancia en el modelo y, probablemente su uso pueda quedar obsoleto. Surge entonces una nueva variante para evaluar el modelo, variar sus predictores. Para ello, utilizaremos métricas diferentes para seleccionar los predictores que mejor separan las muestras en cada nodo, tenemos dos posibilidades, utilizar la medida de curvatura y la medida de interacción, ambas explicadas en el anexo III. Las tasas de acierto calculadas se muestran en la tabla 16:

Métrica de selección	**2-*fold***	**5-*fold***	**10-*fold***
Curvature	0.6739	0.6798	0.6764
Interaction-curvature	0.6664	0.6673	0.6801

Tabla 16. Comparativa de resultados para diferentes funciones de coste

Como, en media, el valor de la métrica de *curvature* es mejor, pero, los resultados no varían respecto al caso anterior, seguiremos empleando el criterio por defecto, CART.

5.3.1. Conjunto de árboles, *Random Forest*

El siguiente paso a dar para crear nuestro clasificador definitivo, es optimizar los parámetros de nuestro *random forest* o conjunto de árboles. Como ya tenemos los parámetros óptimos para cada árbol, el primer parámetro que evaluaremos será el número de árboles. Los resultados de

esta etapa se van a dar para la tasa de acierto, y están recogidos en la tabla 17, aunque en última instancia, la salida del clasificador será blanda.

Número de árboles	**Tasa de acierto sobre datos test**	**Tasa de acierto sobre datos de validación**
50	60.15 %	70.44 %
75	60.88 %	70.70 %
100	60.75 %	70.63 %
125	60.60 %	70.77 %

Tabla 17. Resultados del parámetro número de árboles

Adicionalmente, podemos observar en la figura 39, como varía el error en función del número de árboles entrenados. Sobre la misma, se ha seleccionado el valor de 75 árboles y se tiene un error de clasificación de 0.2934. Si buscamos minimizar el error, encontramos que el valor se situaría en 106 árboles con un error de 0.2914, lo cual es indicativo de que, con nuestra primera prueba, nos bastaría un conjunto de 75 árboles para obtener buenas prestaciones.

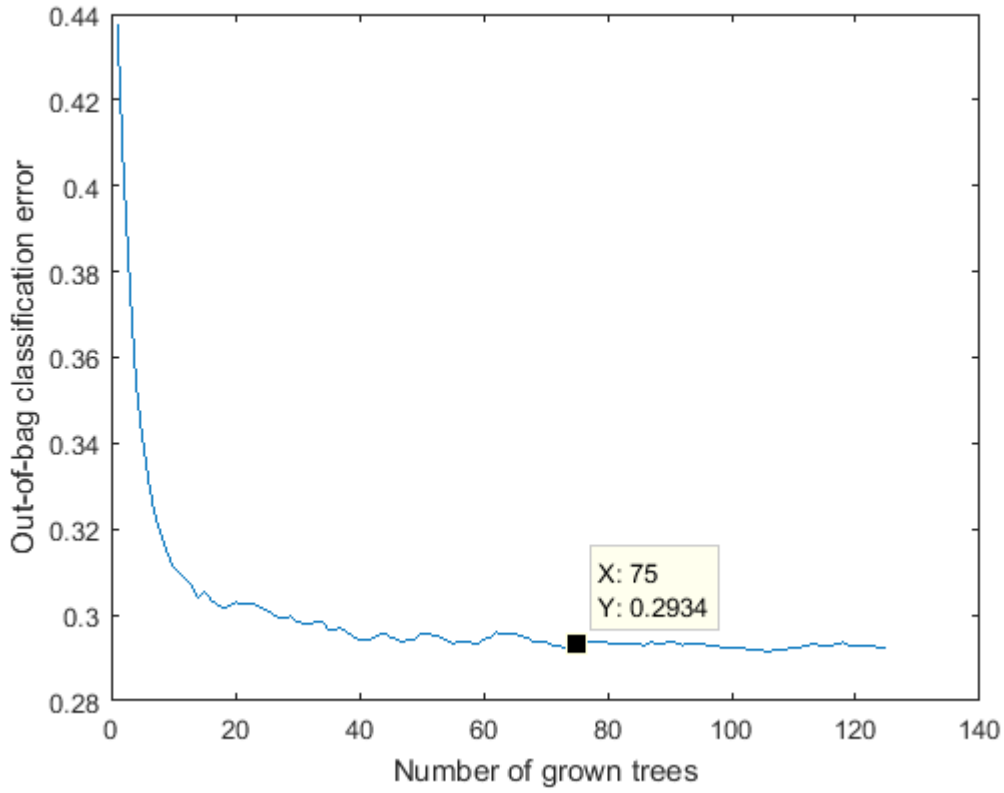


Figura 39. Evolución del error en función del número de árboles

Con estos parámetros de clasificación, podemos pasar al entrenamiento del clasificador definitivo. Para ello, vamos a aumentar el tamaño de nuestra muestra de entrenamiento ya que, como es obvio, la muestra de test va a aumentar ya que es necesario evaluar en cada pixel de cada *frame* para obtener la salida blanda, aunque podremos utilizar un submuestreo de estos.

Al tener en nuestra base de datos un conjunto de 30 videos, dividiremos estos en 5 conjuntos de 6 videos cada uno, con los que entrenaremos los 75 árboles utilizando un 5-*fold* de tal forma que se entrene con 24 videos y se testee con 6. Una vez tenemos nuestros árboles, los añadimos a la "bolsa" o "bosque" y pasaremos a realizar la evaluación final de cada frame, generando un mapa de saliencia para cada plano y evaluando esta con las métricas sAUC y sNSS.

Por último, llevamos a cabo el entrenamiento del clasificador final, utilizando para ello una muestra de dimensión mayor ya que, en este caso, no haremos más pruebas y los datos sobre los que evaluaremos quedarán fuera de la muestra de entrenamiento. Como ya se comentó en el apartado de tecnologías, la salida que tomaremos de la clasificación, no será la clase de cada píxel sino la probabilidad de pertenecer a cada clase, siendo, a la hora de crear el mapa de saliencia, las zonas más salientes aquellas que mayor probabilidad tengan de pertenecer a la clase 1.

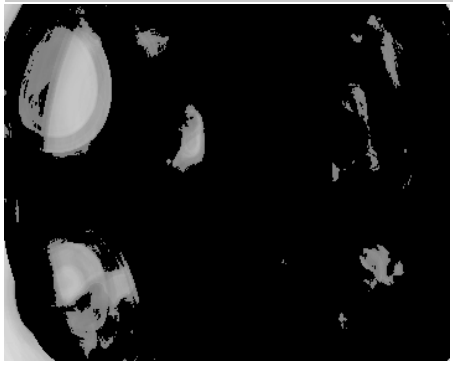
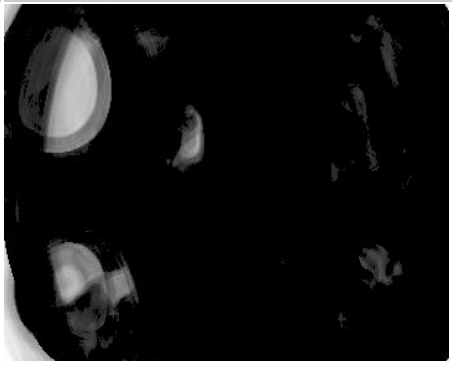
5.4. Evaluación de los modelos de atención visual

Para comenzar la fase final del proyecto y entregar los resultados que nos permitan crear los mapas de atención, tenemos que ajustar como serán estos resultados ya que cada píxel tomará el valor de la probabilidad que tenga de pertenecer a la clase 1, es decir, de atraer la atención visual. Esto, de manera trivial, nos obligas a convertir aquellos píxeles que tienen valor meor que 0.5 en cero, ya que su clasificación es más probable que sea clase 0. Además, para poder ajustar nuestro mapa a un mapa de verdadera saliencia, tendremos que ajustar los valores que resten entre 0.5 y 1 para que tome forma de mapa topográfico y no encontremos saltos directamente de 0 a 0.5. Lo que proponemos para ello, es llevar a cabo una transformación lineal del tipo

$$AttentionMap(x, y) = 2 \cdot (f(x, y) - 0.5)$$

Ecuación 3. Obtención del mapa final a partir de la salida blanda del clasificador

De tal forma que consigamos tener una imagen cuyos valores estén situados en el rango [0,1]. Por último, hemos de llevar a cabo un filtrado paso bajo gaussiano porque, como vamos a muestrear uno de cada cuatro píxeles de la imagen, necesitamos eliminar las zonas espúreas resultados de aumentar la resolución al tamaño original. Los pasos que han sido llevado a cabo, se muestran gráficamente en la figura 40:

1. Mapa con las probabilidades:	2. Mapa tras la transformación:
	

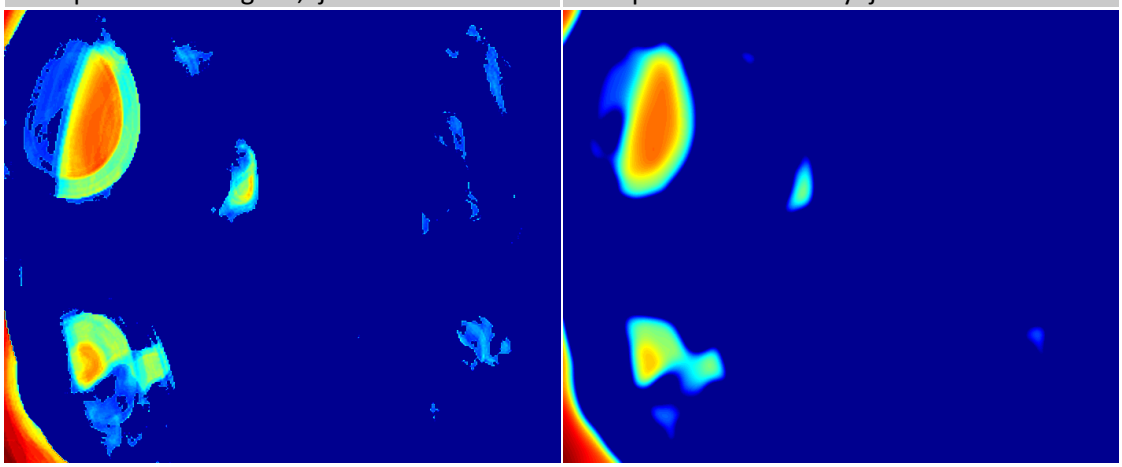
2. Mapa tamaño original, ‘jet’:	3. Mapa tras el filtrado y ‘jet’:
	

Figura 40. Proceso de obtención del mapa final

Como se ha comentado, mediante una validación cruzada 5-*fold*, obtendremos los mapas para todos los vídeos. Usaremos un submuestreo temporal de uno de cada diez planos por razones de tiempo.

Los resultados se expresarán de la siguiente forma, en primer lugar, mostraremos una gráfica con los valores medios de cada métrica por video, en las figuras 41 y 42. En segundo lugar, calcularemos los valores medios totales para nuestra base de datos, y, por último, haremos un análisis por *frames* para el mejor y el peor resultado ya que luego nos ayudará con la discusión de los mismos. El resultado de evaluar los *frames* de la base de datos, generando los mapas de atención para cada uno en base a las *features* optimizadas y el clasificador que hemos utilizado, nos da, como valor medio de sAUC y sNSS para cada video, los siguientes resultados:

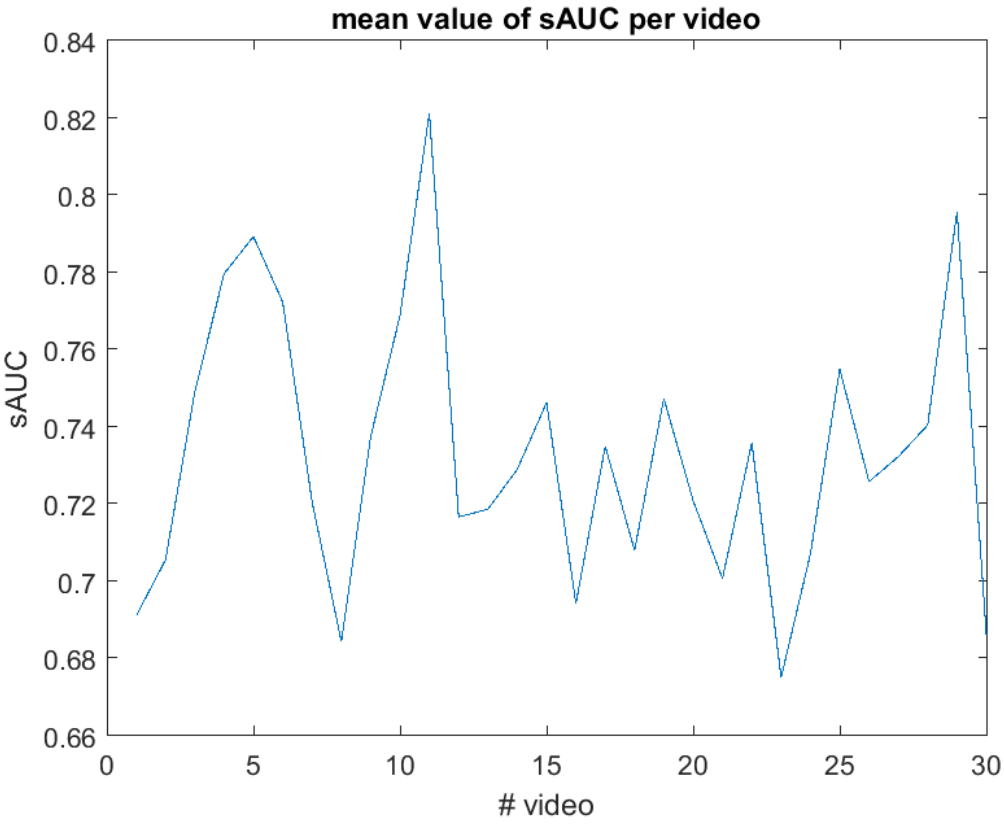


Figura 41. Valor de sAUC medio por video

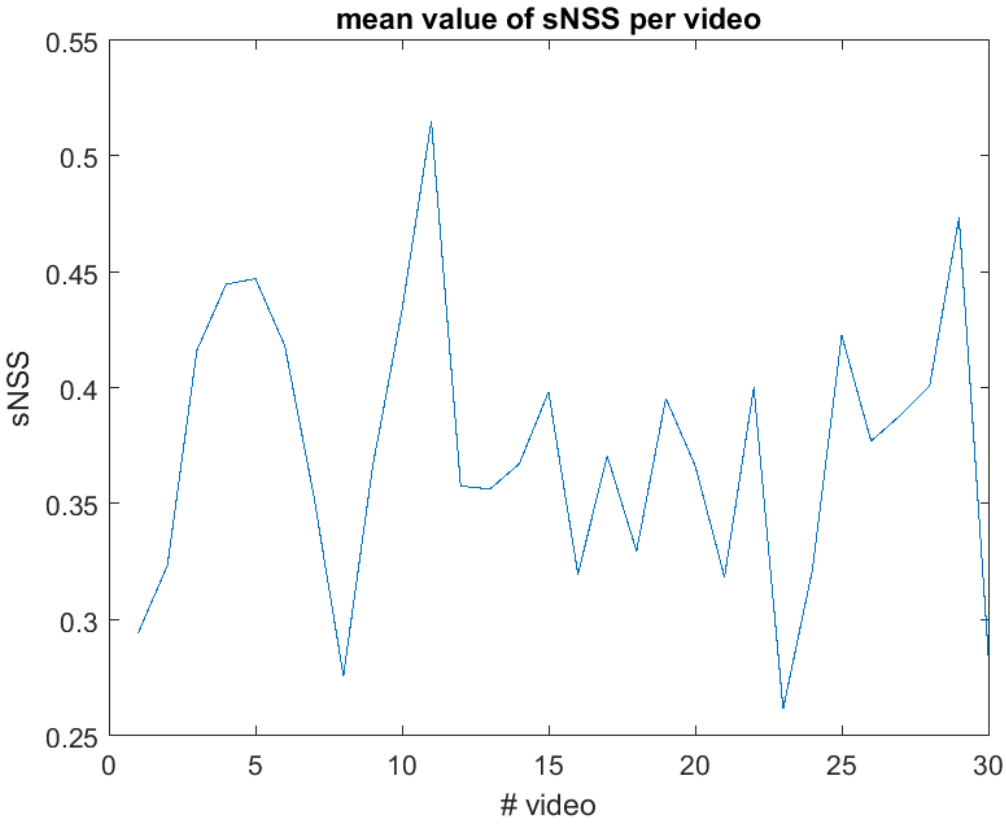


Figura 42. Valor de sNSS medio por vídeo

Con unos valores medios globales indicados en la tabla 18:

valor de sAUC medio	0.7330
valor de sNSS medio	*0.3734*

Tabla 18. Valores medios de sAUC y sNSS por vídeo

Si hacemos una inspección de nuestros mejores resultados y, de los peores, será fácil llevar la discusión sobre los mismos en el siguiente punto. El video cuyos mapas de saliencia han obtenido los mejores resultados tanto para la métrica sAUC como sNSS es el número 11, que corresponde a un video en el que un hombre increpa a una mujer y, los peores resultados, se han registrado en el video 23, que es un video que corresponde a una situación sin anomalía. Por completitud, vamos a realizar un análisis de nuestra base de datos sobre el algoritmo GBVS y el de Itti, para comparar las prestaciones globales del nuestro y, además, analizaremos los resultados que obtendríamos utilizando estos dos modelos para nuestro mejor y peor caso.

5.4.1. Comparación con GBVS e Itti

Si realizamos el experimento de nuevo, utilizando ahora los modelos de Harel[27] e Itti[9], que son los grandes modelos de referencia *bottom-up*, podremos comprobar si un modelo *top-down* que aprende una atención orientada a tareas supera o no a la que se basa exclusivamente en el estímulo. Obtenemos los siguientes valores recogidos en la figura 43 y la tabla 19:

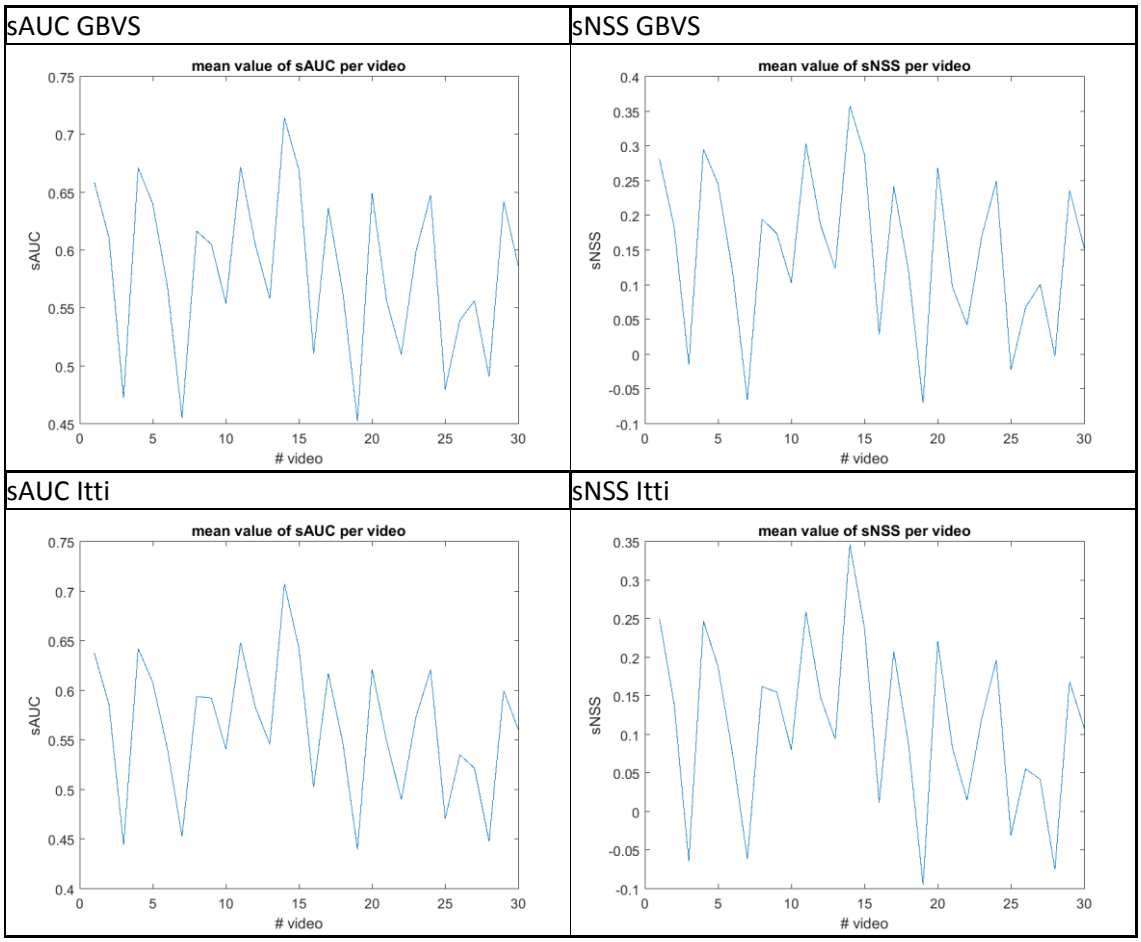


Figura 43. Gráficas de los valores obtenidos con GBVS e Itti

Podemos, finalmente, evaluar en esta tabla los valores medios que han resultado de realizar el experimento con los tres modelos sobre el conjunto de vídeos de nuestra base de datos:

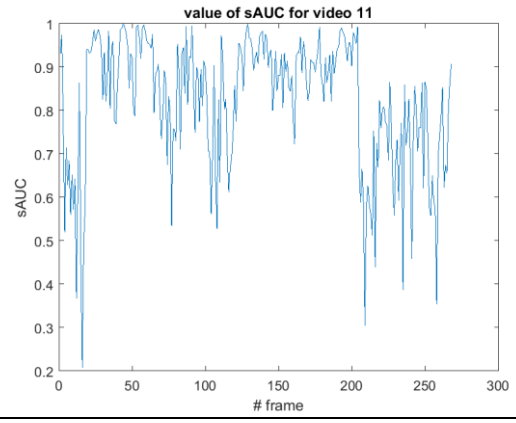
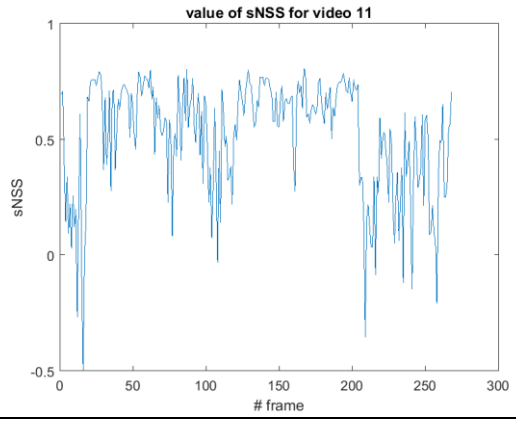
MODELO	**sAUC medio**	**sNSS medio**
top-down	0.7330	0.3734
Itti	0.5622	0.1125
GBVS	0.5831	0.1487

Tabla 19. Comparación final de valores sAUC y sNSS entre los 3 modelos

Con estos resultados tan superiores de nuestro modelo, un 15 % sobre GBVS y un 20% sobre Itti, en cuanto a métrica sAUC y un 25% sobre GBVS y un 25% sobre Itti, en métrica sNSS, podemos establecer la siguiente reflexión.

Los modelos de GBVS e Itti, son los grandes modelos de referencia en atención visual basada en características *bottom-up*. Superar a estos supone una demostración de que aprender con una atención orientada a tareas sobre datos particulares de un ámbito, en nuestro caso la videovigilancia del transporte público, proporciona mejores resultados que aquellos que solo aprenden de los estímulos de la propia entrada. El tanto por ciento en que nuestro modelo mejora a ambos, puede ser tomado como una tasa de aprendizaje sobre el ámbito de aplicación a la hora de usar nuestro modelo, por lo tanto, estaríamos contribuyendo a eliminar la dificultad que tiene este ámbito para los seres humanos.

A continuación, vamos a realizar pruebas más experimentales comparando los resultados de los algoritmos GBVS y el de Itti en nuestros dos casos más extremos, en los que obtenemos el mejor y peor resultado. Esto nos servirá para encontrar mejor las diferencias y extraer mejores consecuencias, en la figura 44, observamos los resultados del primero de ellos:

<table>
<tr><td colspan="2">Máximo: video 11</td></tr>
<tr><td>sAUC modelo propio</td><td>sNSS modelo propio</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Mínimo: video 23</td></tr>
<tr><td>sAUC modelo propio</td><td>sNSS modelo propio</td></tr>
</table>

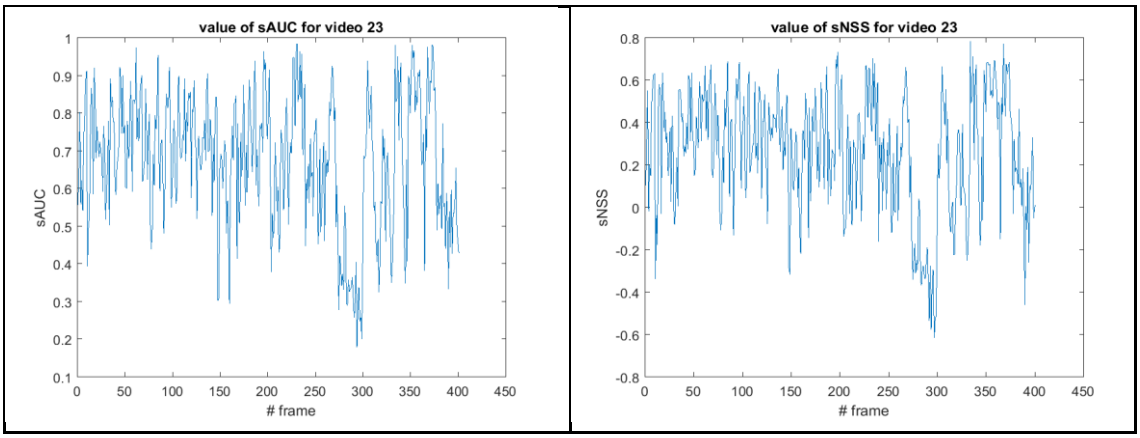


Figura 44. Gráficas de los valores extremos en nuestro modelo

A continuación, se muestra en la figura 45, para el mismo caso y utilizando el modelo GBVS, y en la figura 46 para el modelo de Itti:

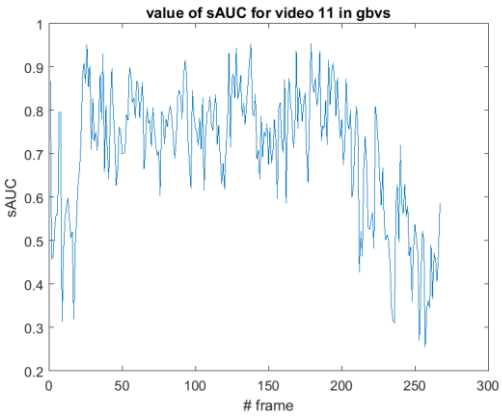
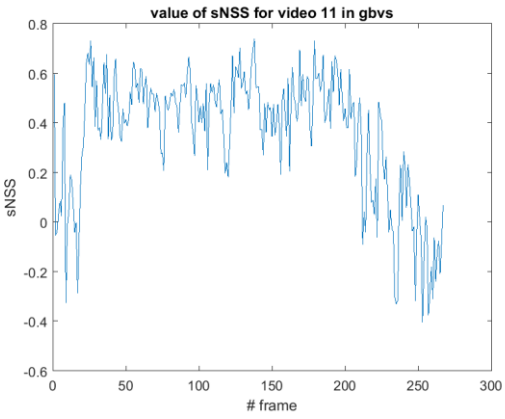
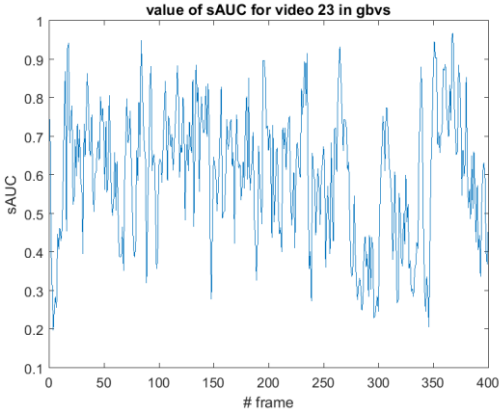
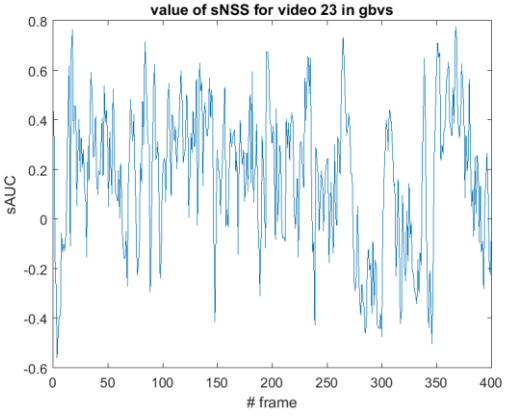
Máximo: vídeo 11	
sAUC gbvs	sNSS gbvs
	
Mínimo: video 23	
sAUC gbvs	sNSS gbvs
	

Figura 45. Gráfica de los valores extremos en GBVS

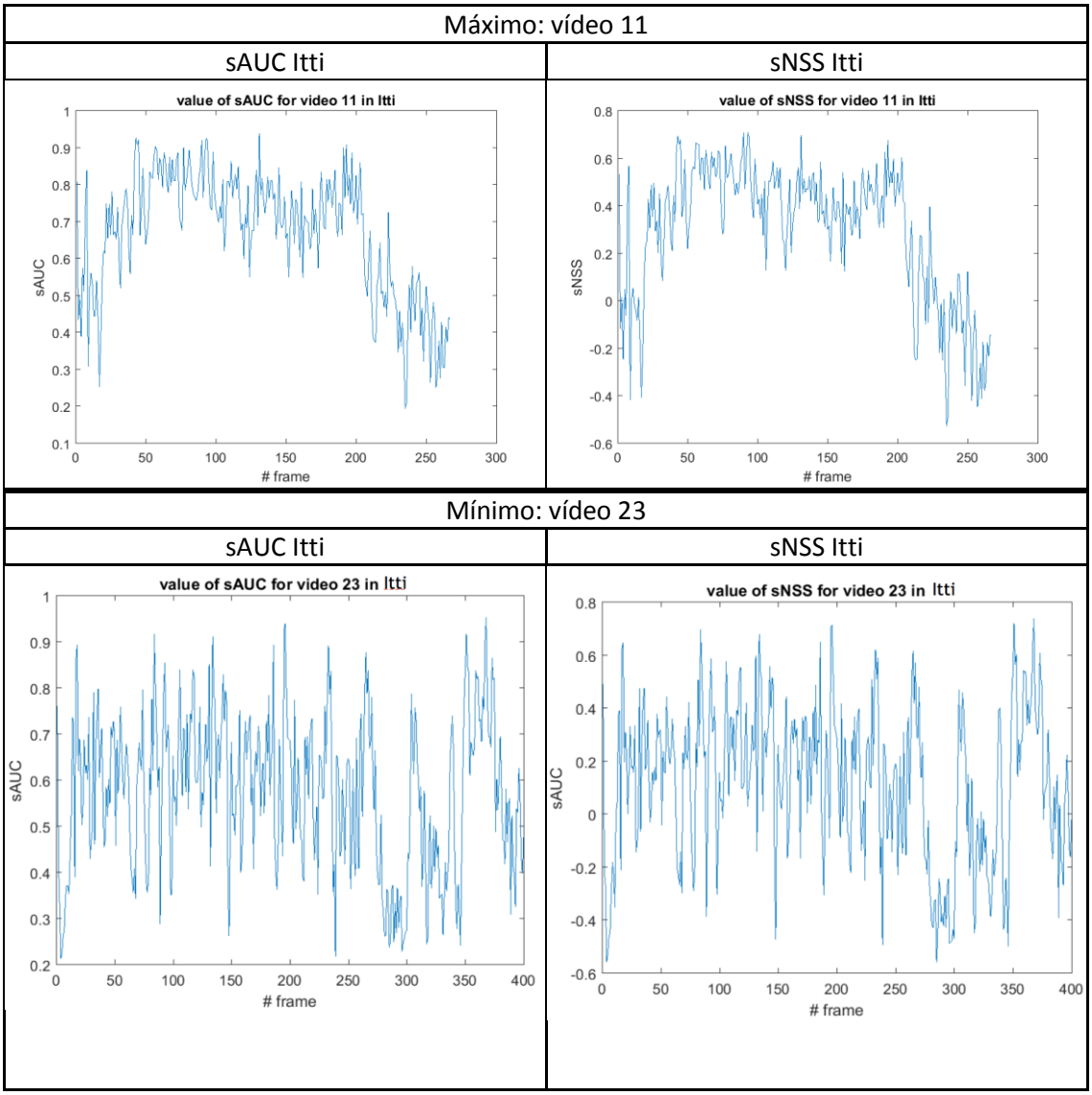


Figura 46. Gráfica de los valores extremos para Itti

Los valores medios para cada métrica y cada vídeo en los dos modelos han sido los que recoge la tabla 20:

número de vídeo	métrica	modelo propuesto	gbvs	Itti
vídeo 11	sAUC	0.8212	0.7074	0.6681
	sNSS	0.5154	0.3559	0.2930
vídeo 23	sAUC	0.6751	0.6000	0.5744
	sNSS	0.2615	0.1701	0.1205

Tabla 20. Comparativa final de valores extremos

Por último, observamos como para los casos más extremos, nuestro modelo también sale vencedor en la comparación. Es importante tener en cuenta este resultado, ya que comprobar, que para nuestro peor caso, que coincide con un video sin anomalía, consigue mejores

resultados que para ambos modelos contrastados. Esto indica, que las características que hemos seleccionado complementan bien el objetivo del modelo.

Podemos, por último, realizar una inspección a nivel de *frame* para tener una idea más precisa de los puntos en los que nuestro modelo supera a Harel e Itti y en los que no. En el video 11, en torno al *frame* número 200, nuestro modelo tiene malas prestaciones, alcanzando valores de sAUC inferiores a 0,5. En ambos casos, los modelos *bottom-up* superan al nuestro, representando el *frame* en cuestión un cambio de situación ya que tanto el hombre como la mujer que estaban simulando una situación de acoso salen del vagón.

Si observamos las gráficas, en los *frames* predecesores, nuestro algoritmo estaba dando mejores prestaciones, ya que en la acción había mucho movimiento y nuestro modelo lo tiene en cuenta, en cambio, cuando la situación se estabiliza y tampoco hay personas que detectar, nuestro algoritmo empieza a fallar más, llevando la atención a zonas erróneas que, sin embargo, en los otros algoritmos no se ven tan desplazadas ya que al menos, los resultados son más estables.

Tenemos, por lo tanto, un punto en el que nuestro modelo tiene opción de mejorar, sería introduciendo otro tipo de características que modelasen la atención cuando no tenemos pasajeros o bien no se producen situaciones anómalas que conlleven mucho movimiento.

5.4.2. Tiempo de procesado

Como se ha comentado en la introducción del proyecto presente, forma parte imprescindible del mismo, asegurarnos de que el tiempo total de procesado, desde que obtenemos un *frame* hasta que generamos su mapa de saliencia, no se exceda de tal forma que la situación anómala pueda no ser detectada a tiempo por el personal encargado. Para tal propósito, hemos de hacer una estimación del tiempo de respuesta medio en el que nuestro modelo presentaría el mapa una vez que el *frame* ha sido registrado. Para ello, hemos medido los siguientes tiempos:

- El tiempo total de cómputo de un mapa es: 15.08 segundos
- El tiempo total de extracción de características es de 8.394 segundos
- El tiempo de predicción, una vez extraídas las características es de 5.48 segundos.

A todo ello, hemos de añadir que estamos trabajando con un equipo casero de apenas dos núcleos de trabajo. La optimización que se puede adquirir en tiempo de computo si trabajásemos en paralelo sería grande ya que el tiempo de extracción de características sería mucho menor, de hecho, deberíamos dividir ese tiempo entre 8. A pesar de ello, si consideramos que se tarda 5 segundos en predecir la salida, estamos actuando bien ya que es un tiempo lo suficientemente corto como para que el personal encargado pueda detectar la anomalía presentada.

5.5. Discusión sobre los resultados

Los valores medios obtenidos para las métricas de sAUC y sNSS son superiores a los que obteníamos cuando tratábamos de optimizar las características que mejores resultados obtenían, magnitud del movimiento y aceleración. Esta comparación no es significativa ni informa sobre si nuestro método ha sido fructífero o no, pero si nos ayuda a esclarecer lo contrario, el método no es malo.

Hemos obtenido unos resultados que se acercan a cotas del 75% en cuanto a métrica sAUC, que simplemente evalúa la tasa de falsos y verdaderos positivos y a un 40 % en cuanto a métrica sNSS, que evalúa la diferencia en cuanto a valor de las zonas con positivos y las que no. Podemos resumir las conclusiones que arrojan estos resultados como:

- La tasa de acierto, es decir, atribuir saliencia a las zonas donde se encuentran las verdaderas fijaciones y no a las que son descartables, ronda el 75% por lo que el modelo tiene un buen acierto y discrimina bien las zonas con fijaciones de las que no.

- Por otro lado, el resultado de la métrica sNSS también es superior al encontrado en la optimización de las características y, el valor en torno al 40% nos informa de que los mapas creados son, más bien, suaves. Este efecto nos lleva a mapas cuyos saltos no son grandes, es decir, no se pasará de un valor cero en un píxel a un valor de uno en el píxel consecutivo. Este resultado es el buscado ya que, recordemos que, a la hora de entrenar nuestro modelo, las zonas próximas a las fijaciones no se evaluaban ya que no se podían considerar como negativos.

Para contextualizar aún más los resultados de nuestro modelo, hemos llevado a cabo una comparación con el modelo GBVS y con el modelo de Itti, que nos ayudará a encontrar mejor aún los puntos fuertes y débiles de nuestro algoritmo.

Los resultados obtenidos por nuestro modelo superan en su valor medio a los de ambos algoritmos testeados y, además, nuestros casos extremos también lo hacen.

Tras los resultados obtenidos, podemos concluir que el punto fuerte de nuestro modelo está en haber añadido como descriptores la magnitud del movimiento, la aceleración del mismo y el detector de pasajeros. Si bien es cierto que este último ha sido el mayor protagonista en la etapa de clasificación, no podemos dar de lado que los otros dos servían como una máscara inicial de gran valor para segmentar las imágenes pues, es obvio, que las zonas con movimiento tienen más probabilidades de centrar nuestra atención.

Sin embargo, para ser justos con el modelo GBVS, hemos de repetir que este es un modelo basado en características *bottom-up* cuya finalidad no está en servir de apoyo a la videovigilancia y es con videos de este tipo con los que lo hemos evaluado. A pesar de ello, la única desventaja que nuestro modelo puede presentar ante ellos, es la necesidad de partir de información anterior, ya que, para calcular las características que implican movimiento, hemos de hacer uso tanto del *frame* del que generamos el mapa de atención, así como del *frame* anterior para obtener una buena estimación de movimiento. Esto, no ocurre en los otros dos modelos y, por tanto, su tiempo de computación a la hora de extraer las características es mejor, pero, no obstante, si nosotros nos situamos en un ámbito que prima por su valor en tiempo real, el necesitar el *frame* anterior para procesar las características no conllevará ninguna limitación porque se supone que trabajaremos en entornos con flujos continuos de vídeos, en los que almacenar un *frame* no supondrá, ni mucho menos, un problema real de memoria

En la figura 47, hemos incluido un ejemplo final con los mapas de atención que genera nuestro modelo frente a los que se generan con los modelos de Itti (parte inferior izquierda) y Harel (parte inferior derecha). En la misma, podemos contrastar nuestra salida (imagen superior derecha), la cual tiene un valor de sAUC igual a 0.75, con el resto y observamos como aquellas zonas en las que se prevé que hay más movimiento están mejor definidas por nuestro algoritmo, lo cual es un punto fuerte de nuestro modelo. Los valores de sAUC para Harel e Itti son, respectivamente, 0.68 y 0.71:

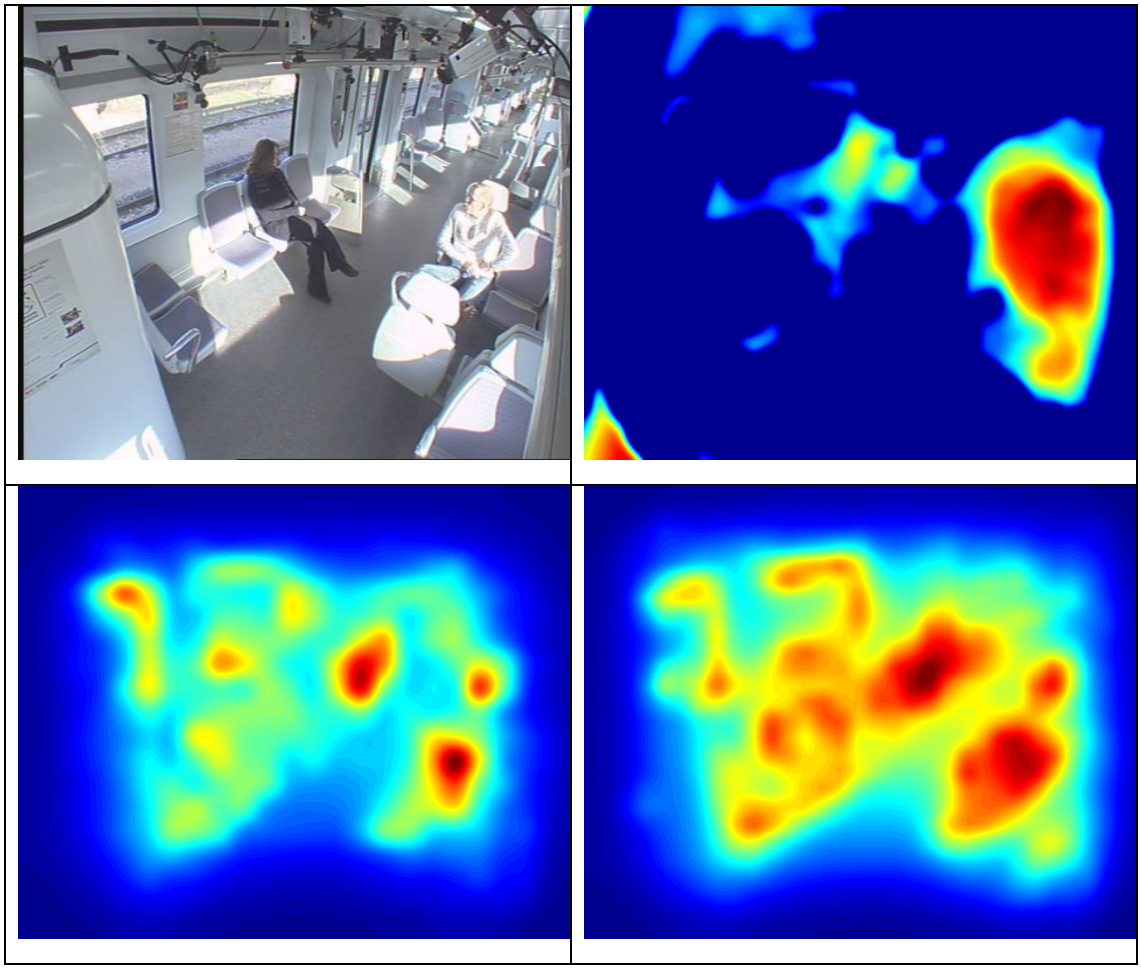


Figura 47. Comparación de nuestro mapa de atención con los de Harel e Itti.

Podemos observar como nuestro modelo, además de escoger mejor las zonas con movimiento, segmenta a los pasajeros, de tal forma que localiza mucho más la atención. Tanto Harel como Itti, proponen modelos mucho más afectados por la centralidad, algo que el nuestro, como se describe en el apartado 4.1.2., no lo hace a través del *center-bias* sino que utiliza el *shuffle map*

Por último, sirva como limitación de este proyecto la poca accesibilidad que existe para obtener videos reales de sistemas de videovigilancia empleadas en el transporte público en el mundo ya que, de esta forma, podríamos evaluar y enfrentar las características globales de nuestro sistema frente a otros.

6. Conclusiones y líneas futuras

6.1. Conclusiones

En el presente documento, hemos desarrollado una base de datos para crear, posteriormente un modelo de atención visual *top-down* como apoyo a la videovigilancia en el transporte público. El modo de creación y desarrollo del mismo, nos permite llevar a cabo una revisión general de cada punto.

La base de datos que hemos creado es, a pesar de los pocos videos de los que disponíamos, una base suficiente para crear un modelo de estas características que nos permita, no solo avanzar en el ámbito de este proyecto sino profundizar en el reto que supone analizar los atractores de la visión humana que se encuentran en este conjunto de imágenes.

Las características que, una vez revisada la base de datos, hemos seleccionado, nos parecen suficiente para la creación del modelo, pero, a modo de algorítmica, una buena combinación de las mismas u otro tipo de procesado de las imágenes, nos hubiese permitido quizá, discriminar mejores zonas de saliencia, pero, no obstante, los resultados avalan el proceso que hemos seguido. Además, cabe destacar que entre nuestras características hemos introducido los tres mapas *bottom-up* que nos ayudan a modelar las zonas que, por su forma, color y textura, también pueden estar llamando la atención en nuestro problema.

La parte que quizá más limite el proyecto sea el clasificador empleado, pues, en un entorno como el actual, con los medios adecuados, quizá incluir tecnología de *deep learning* hubiese supuesto mejores prestaciones en cuanto al modelo global. No quiere esto decir que el *bagging* utilizado no sea adecuado y bueno para nuestro problema, ya que, en este caso, el clasificador es una gran herramienta para el tipo de datos que hemos empleado.

Desde el punto de vista final, con el modelo completamente generado y viendo el trabajo de manera global, queda la sensación de necesitar testear el mismo con muestras de datos mayores ya que, si consiguiésemos construir un modelo cuyas metas estuvieran más definidas, como, por ejemplo, detectar situaciones de acoso sexual, quizás los resultados podrían repercutir en otros ámbitos como el sociológico. No obstante, es un trabajo en el que merece la pena profundizar, aunque sea de una manera como la nuestra.

En el siguiente apartado, analizaremos las futuras líneas de investigación que este trabajo deja abiertas y en las que, de caso de producirse la oportunidad, se debería profundizar para mejorar el mismo.

6.2. Líneas futuras

Una vez desarrollado el proyecto en su totalidad, podemos incidir en tres puntos que han resultado limitantes y que, por lo tanto, deberían poder superarse en futuras investigaciones sobre el tema aquí estudiado:

- En primer lugar, como ya hemos comentado, sería de interés principal mejorar la base de datos con el acceso a archivos de videovigilancia reales más extensos.

- Otra de las mejoras tecnológicas que se deberían implementar serían aquellas que tienen relación con el *deep learning*, entendiendo la complejidad de estas y el poco desarrollo e investigación que se ha producido de las mismas en el campo de la atención visual.

- Una de las líneas futuras más interesante que se debería estudiar es la correspondencia de las fijaciones en videos que se produzcan en el mismo espacio temporal, es decir, videos que, vistos a la vez, reclamen más la atención en unos que otros. Este es el sistema tradicional de los circuitos cerrados de videovigilancia pues, ante una serie de cámaras, todas ellas se visualizan a la vez, distribuyendo la saliencia entre cada una de ellas. Podemos ver un ejemplo de estos sistemas en la figura 48, captada en un supermercado tradicional.

Figura 48. Matriz de videovigilancia de un supermercado

- Por último, la clave del futuro sería tener la posibilidad de evaluar, finalmente, si el modelo desarrollado contribuye a ayudar a los sistemas de detección de anomalías.

7. Estimación de presupuesto

Para finalizar, hemos de llevar a cabo una estimación del presupuesto total que conllevaría la realización de este proyecto, para ello, nos basáremos meramente en criterios técnicos y tendremos en cuenta que existen dos puntos del mismo de excepción: que a los voluntarios no se les pagará ninguna retribución a cambio de visualizar los videos y que los estos serán tomados de bases de datos abiertas publicadas.

Para el desarrollo del grueso del proyecto, haremos uso de tres personas con diferentes cargos cada una:

- Director del proyecto: Será encargado de guiar las ambiciones del mismo y proponer soluciones técnicas que ayuden a la realización del mismo. En última instancia, el proyecto será revisado por él y dará el permiso de su publicación. Su salario estimado debería ser de 20€ la hora.

- Técnico especialista en laboratorio: Su labor se tendrá en cuenta sobre todo en la primera parte del proyecto, participando activamente en la creación de la base de datos y la configuración de la misma. Su salario estimado será de 15€ la hora.

- Desarrollador del proyecto: La labor principal de la que se encargará está persona es de desarrollar y evaluar todas las soluciones propuestas, bien sea a través de códigos de software o análisis matemáticos. El salario estimado a percibir será de 12 € la hora

Adicionalmente, se incluirán como gastos del proyecto estos dos elementos:

- Estación de captación de datos, *Eye tracker*: Se utilizará una base integrada que combina la cámara de seguimiento y un procesador específico con el software necesario para la interacción con la misma. Su coste aproximado es de 15.000 euros, pero, este equipo se comparte con muchos otros proyectos del grupo, por lo tanto, restringiremos su precio como gasto de amortización al 5 %, y su valor será de 650 €.

- Para los días de visualización de videos, se dispondrá de un *catering* sencillo para los participantes en el estudio cuyo coste aproximado será de 15 € por sesión.

En la tabla 21 se hará un resumen de lo anterior y se calculará el coste total:

Concepto	**Coste**	**Unidades**	**Tiempo**	**Total**
Director de proyecto	20 €/hora	1	40 horas	800 €
Técnico especialista	15 €/hora	1	20 horas	300 €
Desarrollador	12 €/hora	1	480 horas	4.760 €
Eye tracker	650 €	1	-	750 €
Catering	15 €	2	-	30 €
			Total:	6.640 €

Tabla 21. Estimación del presupuesto del proyecto

Por lo tanto, con la estimación realizada, el coste global del proyecto ascendería a los seis mil ochocientos noventa euros.

Anexo I: Métricas de evaluación y *shuffle map*

I.1. Métricas de evaluación

Para la optimización de los parámetros utilizados en los descriptores hemos utilizado dos métricas de evaluación empleadas en la evaluación de la saliencia de los modelos de atención visual. Ambas son variaciones a su vez de otras dos: AUC (*Area Under Curve*) y NSS (*Normalized Scanpath Saliency*). La primera de ellas consiste en el área bajo la curva ROC, que utiliza las tasas de falsos y verdaderos positivos para generarla. La segunda, sin embargo, consiste en el valor medio de los valores del mapa de saliencia, evaluadas en zonas salientes y no salientes. Ambas variaciones, que estudiaremos a continuación, difieren por encima de todo en cómo se lleva a cabo el muestreo de los puntos a evaluar.

sAUC[32]

La métrica sAUC que utilizaremos es una medida que combina la tradicional área bajo la curva (ROC), que compara la tasa de falsos positivos frente a los verdaderos positivos; y el shuffle map para llevar a cabo el muestreo de los falsos positivos. Esta medida nace de la centralidad, en media, de las fijaciones y se utiliza para penalizar los modelos que incluyen medidas de *center-bias* como únicos mapas de saliencia. El procedimiento que utiliza para detectar los verdaderos positivos (puntos en los que el mapa de saliencia decide que hay una fijación y coincide con una fijación real, en inglés *true positive, TP*) es el tradicional, a partir del mapa de fijaciones, busca dichos puntos en el mapa de saliencia y en función de los diferentes valores de estos evalúa si hay detección o no. Para ello, en primer lugar, convertimos nuestro mapa de *ground truth* en un mapa que añada un margen a estos puntos de tal forma que no evaluemos negativos en zonas próximas a las fijaciones, en la figura 49 se ve un ejemplo gráfico de esto:

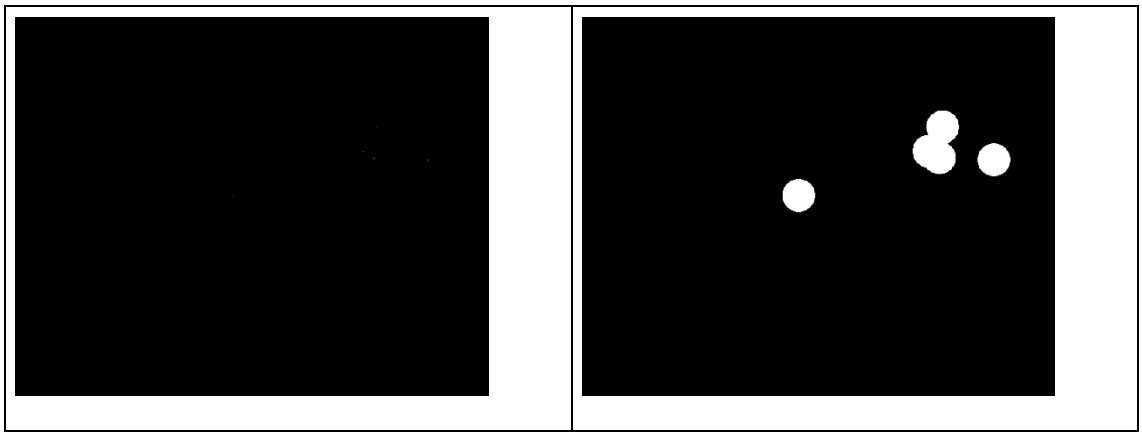


Figura 49. Paso 1, creación de las zonas de guarda

A continuación, sAUC comprueba los valores del mapa de saliencia que están situados en los puntos en los que el mapa de GT sin TH es mayor que cero, es decir, comprueba las fijaciones para calcular el valor que el mapa de saliencia da a estos puntos. La verdadera característica que nos aporta la métrica sAUC consiste en el muestreo de puntos que toma a la hora de evaluar las zonas donde no hay fijaciones, es decir, los negativos de la imagen. Si consideramos que la imagen es binaria, para el caso anterior, sería la siguiente zona mostrada en la imagen de la figura 50:

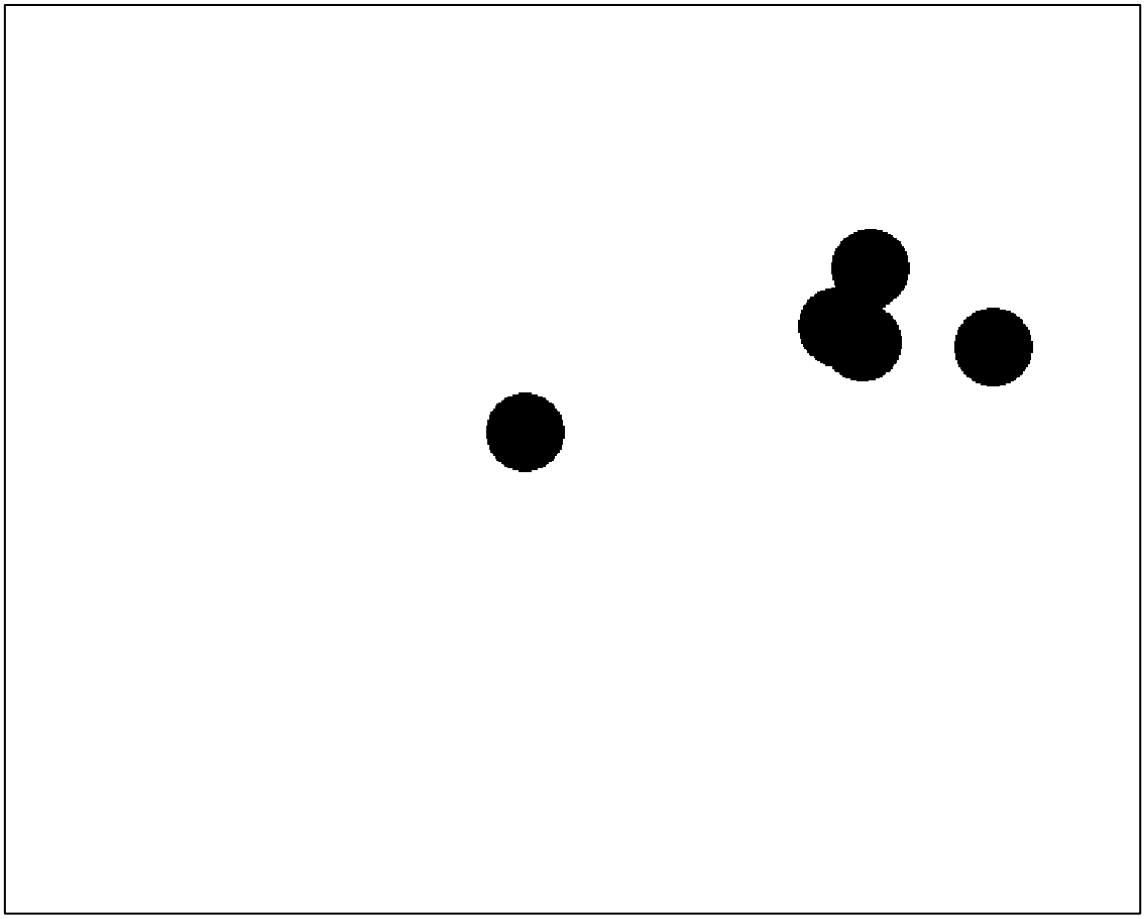


Figura 50. Extracción de los positivos

El muestreo no se realizará aleatoriamente, no se tomará cualquier punto con la misma probabilidad y se evaluará el mapa de saliencia sino que, el muestreo se realizará en las zonas presentes en el *shuffle map* y se tomarán puntos con la probabilidad que el *shuffle map* los asigne, es decir, se multiplican ambas imágenes (punto a punto). Como consecuencia, las muestras de los negativos se tomarán más cerca del centro que en el exterior, penalizando así los modelos que incluyan solo *center-bias* ya que en la zona central de la imagen un modelo de *center-bias* dirá que hay positivos y no será así, ya que las fijaciones no están necesariamente en el centro, pero sí nuestros puntos muestreados. La zona de muestreo de los negativos de nuestra imagen sería la mostrada en la figura 51:

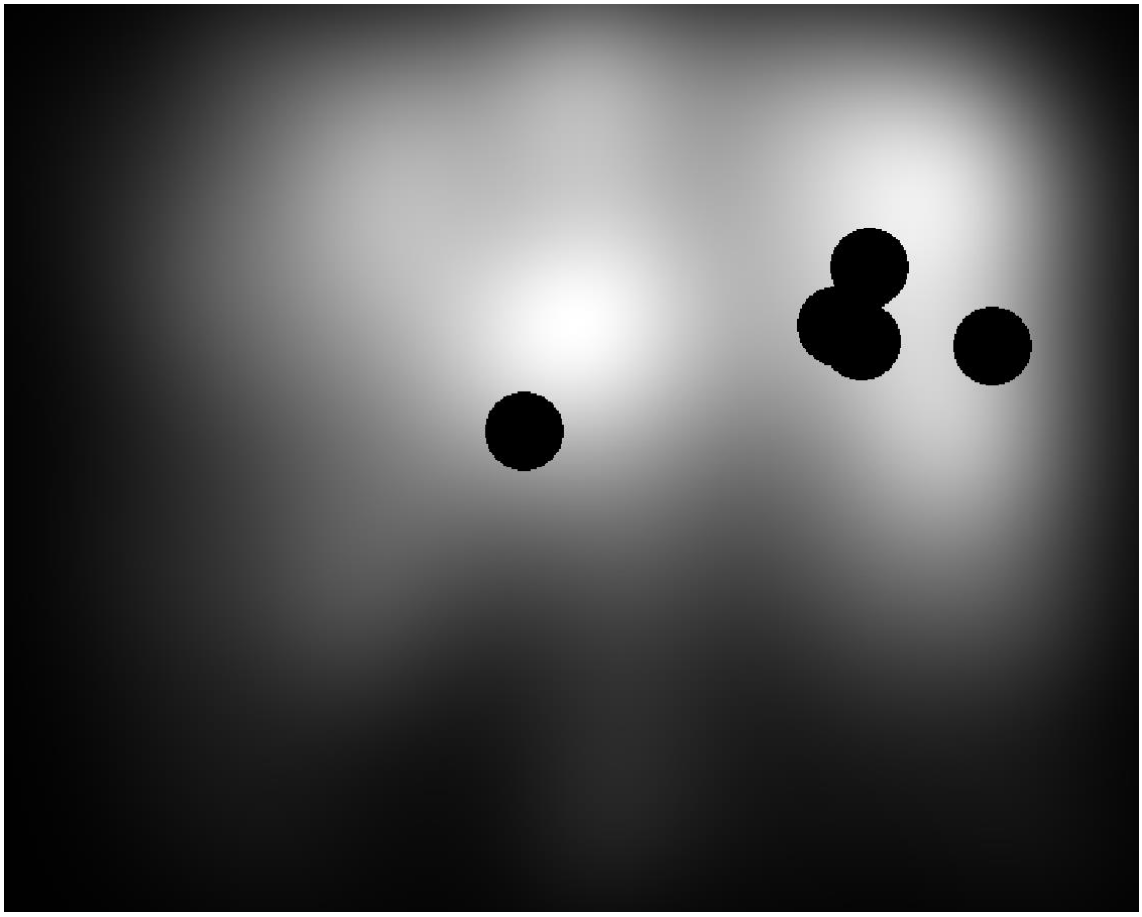


Figura 51. Zonas a muestrear los valores negativos

Además, como hemos dicho, a la hora de muestrear, no solo tomaremos muestras de la zona de la imagen que es mayor que cero, sino que las tomaremos con diferente probabilidad, ya que el *shuffle map* se trata como una distribución de probabilidad.

Por último, el valor resultante de sAUC que obtenemos, nos da información sobre como la tasa de *true positives*, es decir, de aciertos, es respecto a la tasa de falsos positivos. Es decir, que siempre buscaremos un valor alto de esta y trataremos de optimizar los mapas para lograrlo.

sNSS[33]

Esta segunda métrica que utilizaremos nos ayudará a medir la efectividad de nuestros mapas de saliencia en función del valor que estos tomen en sus diferentes zonas, más allá de su efectividad en cuanto a *precisión* y *recall*. En este caso, de nuevo, se modifica la métrica tradicional de NSS añadiéndole un muestreo de negativos basado en las fijaciones previas, es decir, que se empleará el *shuffle map* para seleccionar las muestras sobre las que se calculará este valor.

En términos generales la métrica sNSS normaliza las medidas entre los valores que se atribuyen a zonas con fijaciones y los valores que se atribuyen a zonas de negativos, es decir, que no deben atribuirse a ninguna fijación. Para ello, se realiza una media entre los valores asociados a las fijaciones, sacadas del *ground truth*, y sobre cinco puntos aleatorios muestreados sobre el shuffle map (que excluye las zonas donde tenemos fijaciones) y que atribuye diferentes probabilidades a cada muestra según su localización. A continuación, se calcula la desviación estándar entre dichos valores y por último se normaliza el valor de la saliencia de las fijaciones y se calcula el valor medio de estas. En código, el proceso resultaría:

1. Se calcula la media entre los valores de saliencia de los puntos que corresponden a fijaciones y el mismo número de muestras de la zona negativa:
2. Se calcula, de los mismos valores, la desviación estándar:
3. Se normalizan los valores correspondientes a las fijaciones y se calcula la media de estos, ese será el valor de sNSS para dicho mapa.

De esta segunda métrica, podemos concluir que cuando el mapa de saliencia contenga sus máximos de saliencia en el centro de la imagen, contendrá números falsos positivos y, por ende, el valor de sNSS será negativo. Si, por otro lado, el mapa de saliencia atribuye los mismos valores tanto a zonas de fijaciones como a zonas en las que habría negativos, la media entre estas sería cero y por lo tanto el valor de sNSS también lo sería. Por lo tanto, de nuevo, tendremos que buscar el máximo valor posible de esta métrica a la hora de optimizar nuestros parámetros a usar en la selección de descriptores.

I.2. SHUFFLE MAP

Para ambas métricas presentadas en este capítulo, hemos hecho uso de un mapa de probabilidad para obtener las muestras a evaluar en las zonas que no corresponden a las fijaciones del *frame*. Este mapa modela una distribución de probabilidad que contiene las localizaciones de las fijaciones que ocupan nuestro experimento global, dando distintas probabilidades a los distintos puntos de la imagen en función del lugar que ocupen. El aspecto que tiene un mapa con todas las fijaciones del estudio es el mostrado en la figura 52:

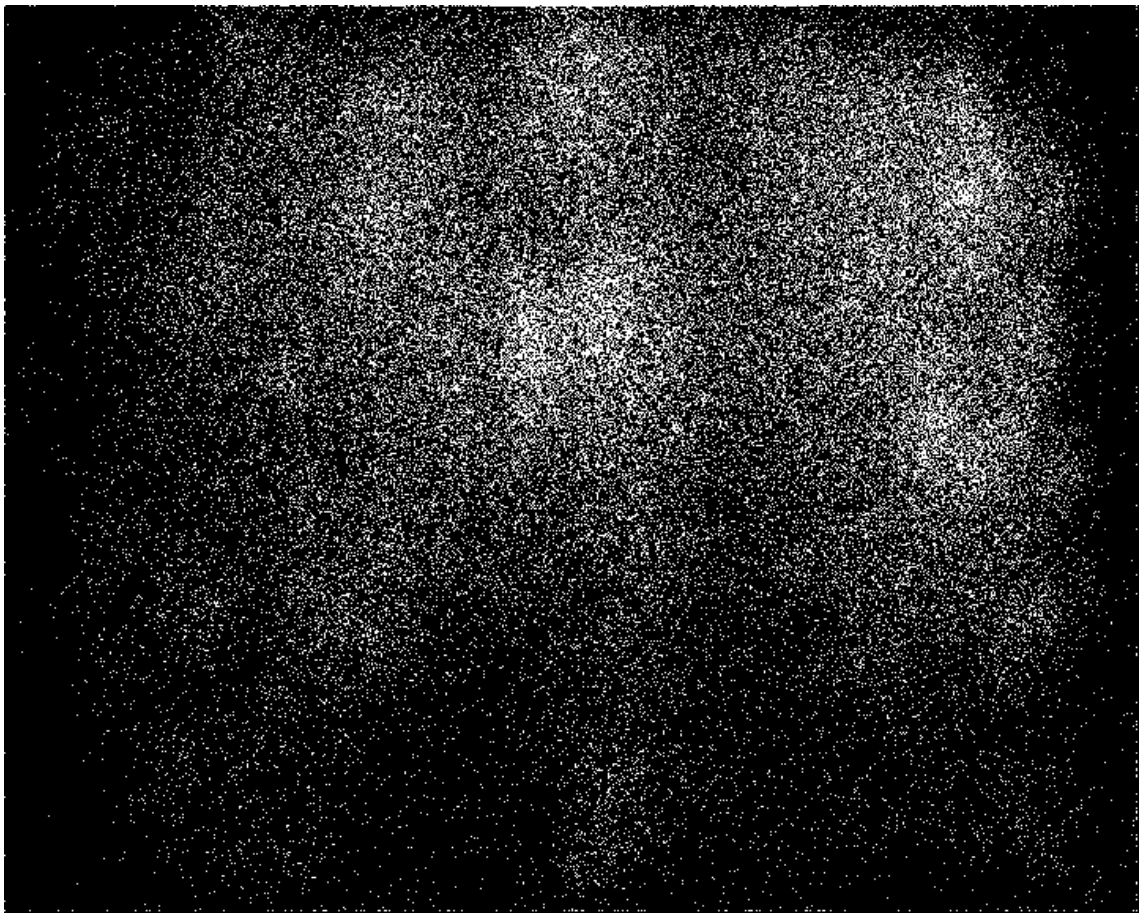


Figura 52. Fijaciones del estudio sobreimpuestas en un mismo frame

El siguiente paso para obtener el *shuffle map* es convolucionar cada *fixation* con una gaussiana con desviación igual a 40 que representa la desviación media a la hora de realizar el estudio, debido a los grados de visión y la distancia al monitor y, a la resolución de la pantalla. Con esto, daremos mayor probabilidad a las zonas con mayor densidad de fixations y viceversa. Por último, como el *shuffle map* debe representar una densidad de probabilidad es necesario forzarlo a que sume uno para el objetivo de este estudio, como se puede observar en la figura 53.

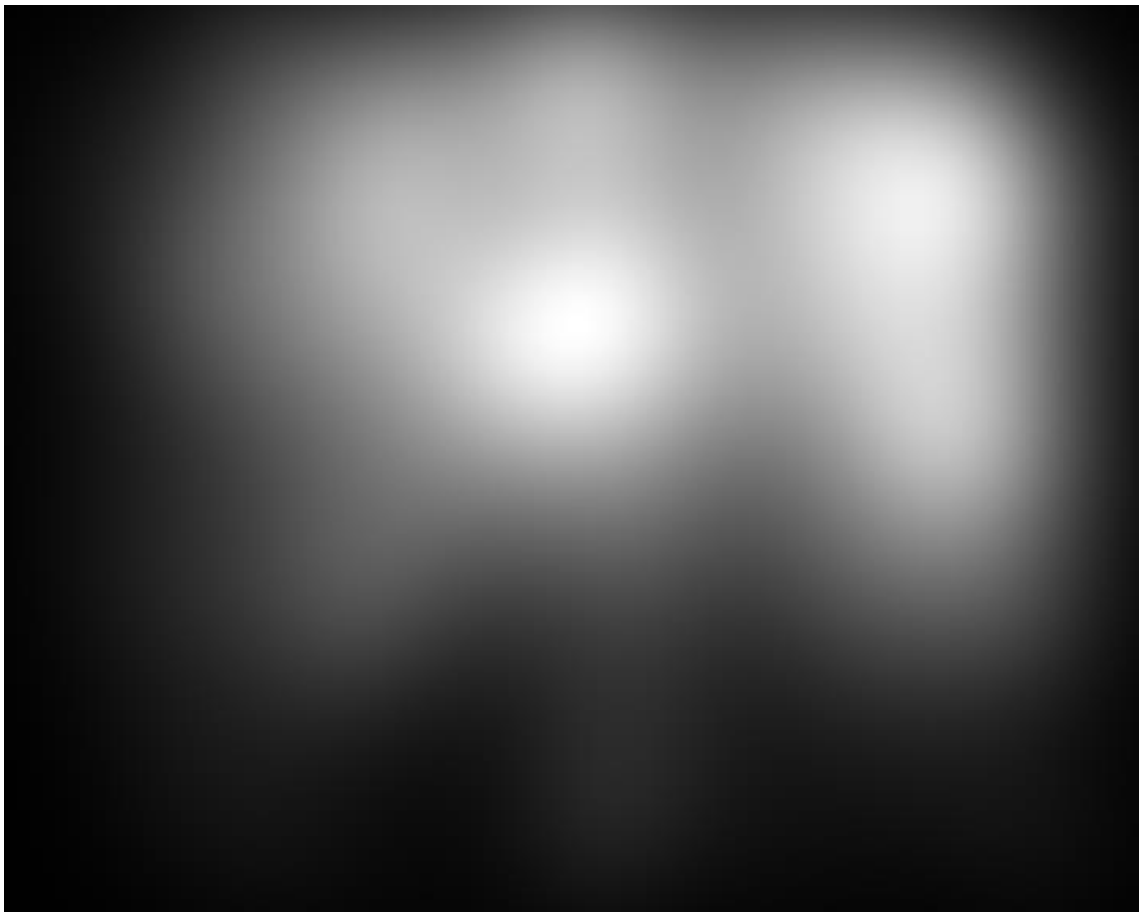


Figura 53. Resultado final de shuffle map

Una vez que el *shuffle map* ya ha sido modelado, tenemos todas las herramientas para comenzar la optimización de parámetros de las distintas *features*.

Anexo II: Descripción del modelo GBVS

II.1. Modelo GBVS

El modelo GBVS[28] viene de las siglas del inglés *"Graph-Based Visual Saliency"* y fue introducido por Harel en el año 2006. Básicamente, el algoritmo consiste en formar mapas de "activación" a partir de los canales de características o *feature channels* y la normalización de estos para combinarlos y remarcar la saliencia.

A continuación, vamos a describir las diferentes partes del proceso, centrándonos en los pasos de creación del mapa de activación y de la normalización y combinación de este.

Formación del mapa de activación:

Partiendo de un mapa de características, F, el objetivo es calcular un mapa de activación, A, tal que la localización (x,y) esté contenidas en este y su valor en el mapa de características, F(x,y), sea diferente del resto de localizaciones del mapa, de tal forma que el valor del mapa de activación para esa localización tendrá que tener también un valor elevado.

Si definimos como medida de disimilitud entre F(x,y) y F(u,v) como la resta logarítmica de sus valores, tendríamos una relación simple entre ambos puntos. A continuación, se construye un grafo completo que conecta cada nodo de F, etiquetado con sus coordenadas, con el resto de puntos de la imagen, asignando un peso a la rama que une el nodo (x,y) con el nodo (u,v) equivalente al producto entre la disimilitud de ambos y un valor que viene dado por la siguiente expresión:

$$P(a,b) = exp(-\frac{(a^2+b^2)}{2\sigma^2})$$

Ecuación 4. Expresión para medir disimilitud

de tal forma que el peso de la rama se definiría como:

$$w_1((x,y),(u,v)) \triangleq d((x,y)||(u,v)) \cdot P((x-u),(y-v))$$

Ecuación 5. Peso para cada rama del grafo

Siendo el parámetro sigma un grado de libertad del algoritmo, Ahora, normalizando todos los pesos del grafo, para que el mayor valga 1, creamos una cadena de Markov de tal forma que los nodos cuyas medidas de disimilitud sea mayor, almacenaran naturalmente, mayor peso.

Normalización del mapa de activación:

El objetivo de este paso es crear zonas concentradas en el mapa que almacenen los máximos valores ya que así, al combinar unos con otros, se conseguirá un "*master-map"* que concentre la saliencia en determinadas localizaciones, siendo así todo lo contrario a un mapa uniforme y carente de información. Para normalizarlo, se propone otro algoritmo Markoviano que vuelve a introducir el concepto de grafo, dando a cada rama el siguiente peso:

$$w_2((x,y),(u,v)) \triangleq A(u,v) \cdot P((x-u),(y-v))$$

Ecuación 6. Peso de cada rama del mapa normalizado

Y, de nuevo, normalizando el valor de todas las ramas podemos tratar al resultado como cadena de Markov que se puede tratar como una distribución de masa, de tal forma que los nodos con mayor masa sean zonas más salientes.

Finalmente, podemos mostrar un ejemplo de cómo funciona este algoritmo a través de la figura 54. En la primera, observamos una imagen en la que se han seleccionado un número de puntos con cruces amarillas correspondientes a fijaciones humanas. Abajo a la izquierda, tenemos un mapa de saliencia que propone el algoritmo GBVS y a su derecha, un mapa tradicional basado en el *center-bias* y diferencia entre gaussianas. Además, se incluye el valor de área bajo la curva de cada uno de ellos.

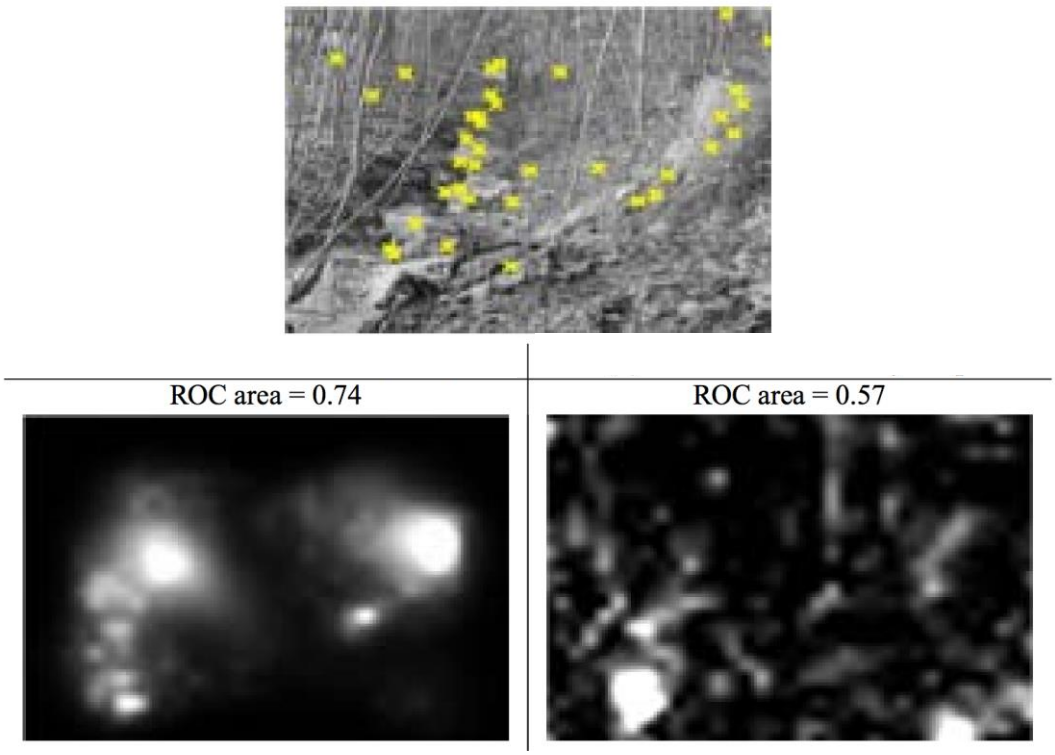


Figura 54. Ejemplo de dos mapas de saliencia, a la izquierda, un modelo tradicional, a la derecha, GBVS

Anexo III: Parámetros de un árbol de decisión

III.1. Índice de Gini, entropía, test de curvatura e interacción

Índice de Gini[35]:

Dado un nodo del árbol de clasificación, el índice de Gini mide la pureza del mismo como la probabilidad que existe en un nodo de no etiquetar dos muestras en la misma clase. Siendo valores altos de este propio de nodos impuros, mientras que valores bajos, significaran que el nodo divide la muestre en una clase mayoritaria y otra de menor tamaño, es decir, es un nodo puro. El índice de Gini se calcula como:

$$Gini(nodo_i) = 1 - \sum_{c \in C} p_{t,c}^2$$

Ecuación 7. Fórmula del índice de Gini

dónde *C* denota el conjunto total de clases y *p* es la probabilidad de ocurrencia de la clase *c* en el nodo *i-ésimo.* Es decir, que si tenemos dos clases, 0 y 1, la probabilidad de ocurrencia de la clase 0 será igual al número de muestras de la clase 0 que se encuentran tras la división del nodo, dividido entre el número total de muestras que clasifica el nodo.

Entropía muestral:

La entropía es una medida de homogeneidad de una muestra. Si la muestra es completamente homogénea, i.e., contiene muestras de una sola clase, el valor de su entropía es zero y, por el contrario, si la muestra está igualmente dividida entre el número de clases total, su entropía es uno. La entropía de un conjunto de datos, *S*, viene dada según la siguiente expresión:

$$Entropia(S) = \sum_{i=1}^{c} - p_i \cdot log(p_i)$$

Ecuación 8. Expression para calculary la entropía muestral

Dónde *c* designaría el número total de clases del problema y p_i es la proporción de valores de la clase *i* dentro de la muestra.

Además, la medida de entropía nos puede ayudar a establecer criterios tanto de parada como de separación. Como criterio de separación, podemos utilizar la ganancia de información. La ganancia de información para una característica F, se calcula como la diferencia de entropía de la muestra antes de la división y posterior a esta. También, el hecho de trabajar con medidas de entropía, nos permite establecer un valor mínimo de esta para establecer un criterio de parada de tal forma que, cuando se supere el valor mínimo, se pueda considerar al nodo como un nodo hijo.

Curvature:

El test de curvatura es un contraste de hipótesis estadístico que trata de evaluar la hipótesis nula de que dos variables están asociadas. Para llevar a cabo este test, se van seleccionando una a una las variables predictoras y se asocian con la salida. En nuestro caso, como los predictores son variables continuas, las dividimos en sus cuartiles y se estudia la proporción de clases que tendría cada uno. A continuación, se construye el estadístico de prueba para realizarle el test *chi-cuadrado* y finalmente, en función del *p-valor* obtenido se toma una de las siguientes decisiones:

- Si todos los p-valores son mayores que 0.05, el nodo no se particiona.
- Si entre los valores, hay uno que está por debajo de 0.05, se elegirá el predictor correspondiente para dividir este nodo.
- Si más de un p-valor es menor que 0.05, se toma una decisión predefinida para el caso, ya que esto puede ser debido a una característica intrínseca de los datos

Interaction- curvature[36]*:*

Esta segunda métrica de medición del error aplica sobre la anterior el test de interacción. Esta prueba evalúa la hipótesis nula de que no existe interacción entre un par de variables predictoras y la variable de respuesta. Para ello, toma en cuenta las siguientes consideraciones:

1. Particiona una de las dos variables predictoras candidatas en sus cuartiles, creando una nueva variable nominal para cada partición.
2. Se realiza el mismo proceso con la otra variable y se realiza una transformación de tal forma que se crea una variable auxiliar como el producto de cada cuartil de ambas variables.
3. Se realiza un test de curvatura, descrito en el apartado anterior, para la variable resultado y la variable respuesta.

De nuevo se realizan los mismos test que en el caso anterior para decidir si el nodo se divide y con qué atributo lo hace.

Anexo IV: Extended summary

The increase of technology in areas like human's vision allows to create new models of visual attention task-driven.

In our report, we'll try to create one of them based on the field of video surveillance in public transport, trying to analyse the spaces in the images that, appreciably, attract our visual attention when we face this task. To make that, we'll use a set of classification trees because we need a performance that make the prediction result fast and appropriate to the model.

In addition, it will be a main part of this project, the creation of a dataset that allows us to develop the system. This dataset should be flexible and specify because there aren´t so much free datasets and we should train our model with a few videos. Our goal is to infer a model which, form our sample of videos, could apply in the best way to predict in some other videos of the same field. So, we must generalize the model in all his process to obtain better results in other scenes.

Finally, we need to compare our results with the results that obtains come other specific models when they process our videos, so we can make us an idea of where our algorithm is good and which are the points we should improve.

With the evolution of public transport, the security's systems that are integrated in them, have undergone a parallel evolution, improving and looking for the last technological possibilities in the market. This fact, has directly affected the introduction of video surveillance's circuits in almost any public transport on the planet since it is also a function of these, to guarantee the safety and experience of the users.

This situation places us in a scenario with immense capacities for the image and video's analysis and process, since constantly and throughout the time, images are being captured that are no longer relevant till the moment in which a passenger suffers, fortuitously or caused, an anomalous situation. In our dataset, we are going to study six types of these events:

1. Mobile phone's theft: in these videos, we observe a theft of a mobile phone between two people.
2. Fight: This sequences shows us a fight between two or more travellers in a carriage.
3. Disease: We could watch in this video a situation of a person falling down because an illness. When this happens, the rest of the passengers in the train go quickly to him.
4. Harassment: That set of videos simulates situations of harass from a man to a woman. When it happens, the woman runs outside the carriage to make it more visual.
5. Newspaper's row: In this sequences, two people argument because they both want to read the newspaper and could go into a fight.
6. Panic: We've got videos in which a person spread terror and the rest of the people in the train run outside because they think that maybe it's a terrorist attack.

Additionally, we've got too some other videos in which nothing is happen. That's because we need to train our model with the most information as is possible, and the model should be realistic so we must add situations with no anomalies in them.

We're confronted with videos recorded with fixed cameras that capture movements of people that, depending on the situation that happens, will claim to a greater or lesser extent our attention.

In pursuit of this, a new challenge for science arises: to reduce the storage space and processing time, so that images captured by the video surveillances systems maintain the utility and improve its efficiency. We need to have fast systems that operate in real time, so that why their response time must to be minimum. We should help security authorities to act in time to the possible anomalies caused.

The way to improve that proposes this project is to create an automatic system that supports the video surveillance in the public transport. To do this, we must properly select the technology to be used.

The classification methods have to be effective and fast due to the need of respond in real time, for this, we will use classification trees and Random Forests, in addition, we will include in our model features that allows us to extract the nature of our database, taking particular care with the size of the sample that we handle since, within the scope of the video surveillance is not too easy to obtain access to videos of companies, as we explain in the section dedicated to the regulation in the chapter 1, section 3, of this project.

The reaction to the stimuli that humans possess has always been defined within the nervous system. System which is dominated by the brain, organ where the stimuli arrives and from which the responses are generated. This system is the result of years and years of evolution, in which the essential parts of our specie have been characterized and which now place us in a stage of research and technological development that, more than ever, focuses on being able to imitate the behaviour of the most perfect machine that ever has existed, ourselves.

It is the nature of these answers so precise and that so little time supposes us to take, the basis of our study, because, in the visual field, attention is limited. This limitation is that, when facing an image or a video we go, directly, to the areas that interest us, areas where we find the information appropriate for the task we face (in our project, to detect anomalies situations in the public transport) and, if we haven't a specific task, areas that are attractive to us and those that we say that "grab attention".

Given the growth of digital information today, the processing of generated data, becomes an impossible task for humans, having to create new computers that allow us to perform these tasks. But, even so, these systems also have limitations when processing the huge amount of bytes, having to resort to the already famous cloud. In short, having a workspace that allows us to store and process the "big data" is to assume a set of expenses in terms of time, work and money.

In our case, we will refer to the field of video surveillance in public transport. In it, we have a task to develop, that involves the use of multiple cameras that must be processed by operators who, as we have said, will have limited visual attention. This fact, leads to high response times that the automatic analysis of the same could reduce, but, on the other hand, the facto of having immense amount of information would lead to increased computing time, thus allowing some non-event were detected, being lost in the great tangled mess of data.

Visual attention systems allow the filtering of such information, making easier the task to later systems of analysis whose use would be restricted to areas marked by interest by the former.

It is understood that our objective model will be the first link in a chain that allows us to take a step in the processing of large masses of data to achieve optimum efficiency of the same. In addition, we cannot lose sight of the fact that our function will also be to develop a fast response

model, so we will use classification algorithms that have, among their strengths, a low computational time when classifying each data arrives. Decision trees and Random Forest could be perfectly located in this type.

The main objective of this project is to create a model of visual attention based on characteristics of top-down type that serves as support in public transport's video surveillance. For this, we'll study different algorithms and metrics that leads us to give a better solution to the database that will be used as a source of information for the project, but, with specially attention to avoid the overfitting in our model.

Among the objectives that we find by conducting an in-depth analysis are:

- Create a database of fixations that tag each of thirty videos used for the development of our model.
- Analyse the fixations in such a way that we could find out the ideal features to extract them of each frame. And then, optimizing these in such a way that their use gives us the best benefits.
- Obtain a classification model that generates saliency maps that meet expectations for each frame.
- Evaluate the ability of decision trees and Random Forest to generate models based on top-down visual attention features, since these have the advantages of having short classification and prediction times that would suit us well when applying the model to real situations of video surveillance.
- Integrate the different parts of the model so the processes are coherent and adequate.
- Be aware of the evaluation metrics that should be used in such problems.

In addition, the following point may be included as a pre-project objective because, within it, we aren't being able to star the study:

- Know the operation and configuration way of the eye tracker in such a way that we can use it with total guarantees for the achievement of the previous ones.

To obtain those objectives, we're going to structure the report in a way that we could follow the steps by this project is created. In first time, we describe the introduction of the project, including a socio-economic contextualization and the methodology used. In addition, we are going to revise the actual regulation framework. The next step is to make a revision of the actual state of art, making special interest in the models of visual attention task-based. Once we introduced everything, we start creating the database, detailing the files and videos that have been used for its creation. We will distinguish it in two phases: configuration and capture.

The main part pf the research is the justification of the proposed solution. We've done a revision of the features used, the sampling process apply, and the final classificatory algorithm used. For optimize all of them, we must make some experiments that give us the results for discuss and compare with some other visual attention models.

The descriptors used can be divided into two groups, according to the stimuli of the scene, ie the design of the same, we have characteristics of the bottom-up type, which correspond to the salience maps of the GBVS model, which we find a development in the Annex; And, according to the task in question, we will find characteristics that consist of the movement of the image and the detection of passengers. In addition, we introduce the central bias of the vision so that the centrality of the fixations in average is collected. Brainstorming proposes a series of points that we must analyze to get the descriptors:

- The movement of passengers on the train is a cause of fixation in situations such as an empty train or sitting passengers quasi-static, therefore, we will have to study the optical flow to find the areas where movements occur. The nature of the videos creates a problem for us through the windows, since the changes of light will cause movement and are not areas of visual attention, so we must study how to discriminate them.

- The acceleration of the movement itself causes fixations because when a passenger moves "abruptly" is a good source of information since these movements can often derive in anomalous situations (thefts, fainting, aggressions, etc ...) although, it could also be reason for a change of total illumination, for example, the exit of a tunnel.

- Fixations are produced on average in the center of the image, so it will be convenient to study the center-bias as a feature, but, our database contains videos with passengers sitting in the seats of the train, which are in the side and, therefore, action, will not always occur centered.

- The faces of the passengers are a clear reason for fixing, in fact, when a passenger is located, attention will always be taken to the facial area, being able to go to other points according to the movement of these, such as hands and legs. Behind these ideas, we must give them body in such a way that we find descriptors of the images that model them. First, we must use a motion estimation algorithm, both to calculate the magnitude of the motion, and to calculate the acceleration itself. Secondly, we will also have to implement a face or passenger detector that provides good results.

Finally, the last step of the project is to have conclusions. The conclusions must to be based in our advantages and defects by comparing our results and our times of computation with the other models.

In this document, we have developed a database to create a top-down visual model to support video surveillance in public transport. The way of creation and development of the same, allows us to carry out a general review of each point.

The database we have created we have created is, despite the few videos we had, a sufficient basis to create a model of these characteristics that allows us not only to advance within the scope of this project but also to deepen the challenge analyse the attractors of the human vision found in this set of images.

The features that, once reviewed the database, we have selected, we seem sufficient for the creation of the model but, as an algorithm, a good combination of the same or another type of processing of the images, would have allowed us to discriminate better areas of salience but, nevertheless, the results endorse the process that we have followed. In addition, it should be noted that among our features we have introduced the three bottom-up maps that help us to model the zone that by their shape, colour and texture may also be drawing attention to our problem.

Perhaps, the most limiting part of the project is the classifier used, because in an environment like the present one, with the right means, perhaps including deep learning technology would

have meant better performance in terms of global model. This does not mean that the bagging used is not suitable and good for our problem, because in this case, the classifier is great for the type of data we have used.

From the final point of view, with the model completely generated and seeing the work in a global way, there remains the feeling of needing to test the same with larger data samples since, if we could construct a model whose goals were more defined, if we identify the anomalous situations we are encountering in a single instance, such as detecting situations of sexual harassment, perhaps the results could have repercussions in other areas such as sociological.

However, it is a job that is worth working even it is in a way like ours.

Once the project has been fully developed, we can focus on three points that have been limiting and that, therefore, should be able to be overcome in future research on the subject studied here:

- Firstly, as we have already commented, it would be of special interest to improve the database with access to more extensive real video surveillance archives.
- Another technological improvement that should be implemented would be those related to deep learning, understanding the complexity of these and the little development and research that has been produced in the field of visual attention.
- Finally, the most interesting future line that should be studied is the correspondence of the fixations in videos that occur in the same time space, that is, videos that, seen at the same time, demand more attention some than others. This is the traditional system of closed circuits of video surveillance because, facing a series of cameras, all of them are visualized at the same time, distributing the salience between each one of them.

REFERENCIAS

[1] Rosselló Mir, Jaume, Munar Roca, Enric, Resolviendo el puzzle de la atención visual: ¿Hacia la desintegración del «homúnculo»?Psicothema [en linea] 2004, 16

[2] Deubel, H, and W X Schneider. "Saccade Target Selection and Object Recognition: Evidence for a Common Attentional Mechanism." *Vision research* 36, no. 12 (1996): 1827-37.

[3] Cisco, V N I. Global IP Traffic Forecast, 2013-2018. 2014.

[4] NORRIS, Clive; MCCAHILL, Mike; WOOD, David. The Growth of CCTV: a global perspective on the international diffusion of video surveillance in publicly accessible space. Surveillance & Society, [S.l.], v. 2, n. 2/3, sep. 2002. ISSN 1477-7487.

[5] "Ley Orgánica 15/1999, De 13 De Diciembre, De Protección De Datos De Carácter Personal." *Bol Del Estado* 298, no. 2 (1999): 43088-43099.

[6] Borji, Ali, and Laurent Itti. "State-of-the-art in Visual Attention Modeling." *IEEE transactions on pattern analysis and machine intelligence* 35, no. 1 (2013): 185-207.
[7] Anne M. Treisman, Garry Gelade, A feature-integration theory of attention, In Cognitive Psychology, Volume 12, Issue 1, 1980, Pages 97-136, ISSN 0010-0285,

[8] Niebur, Ernst. "Saliency Map." *Scholarpedia* 2, no. 8 (2007): 2675.

[9] Itti, L, and C Koch. "A Saliency-based Search Mechanism for Overt and Covert Shifts of Visual Attention." *Vision research* 40, no. 10-12 (2000): 1489-506.

[10] Legge, G. E., Klitz, T. S., & Tjan, B. S. (1997). Mr. Chips: An ideal-observer model of reading. *Psychological Review, 104*(3), 524-553.

[11] Salvucci, D. D. (2001). An integrated model of eye movements and visual encoding. *Cognitive Systems Research, 1*(1-4), 201-220.

[12] I.A. Rybak, V.I. Gusakova, A.V. Golovan, L.N. Podladchikova, N.A. Shevtsova, A model of attention-guided visual perception and recognition, In Vision Research, Volume 38, Issues 15–16, 1998,

[13] A. Oliva, A. Torralba, M. S. Castelhano and J. M. Henderson, "Top-down control of visual attention in object detection," *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, 2003, pp. I-253-6 vol.1.

[14] Rolfs, Martin. "Attention in Active Vision: A Perspective on Perceptual Continuity Across Saccades." *Perception* 44, no. 8-9 (2015)

[15] Benfold, Ben, and Ian D Reid. "Guiding Visual Surveillance by Tracking Human Attention." In *BMVC.* 2009.

[16] María T. López, Antonio Fernández-Caballero, Miguel A. Fernández, José Mira, Ana E. Delgado, Visual surveillance by dynamic visual attention method, In Pattern Recognition, Volume 39, Issue 11, 2006, Pages 2194-2211, ISSN 0031-3203

[17] L. Maddalena and A. Petrosino, "A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications," in *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1168-1177, July 2008.

[18] Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet

[19] Weiming Hu, Tieniu Tan, Liang Wang and S. Maybank, "A survey on visual surveillance of object motion and behaviors," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 3, pp. 334-352, Aug. 2004.

[20] Hoja de especificaciones SMI iView RED250

[21] L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers - a survey," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 4, pp. 476-487, Nov. 2005.

[22] Raileanu, Laura Elena, and Kilian Stoffel. "Theoretical Comparison Between the Gini Index and Information Gain Criteria." *Annals of Mathematics and Artificial Intelligence* 41, no. 1 (2004): 77-93.

[23] Sepúlveda, Juan Felipe Díaz, and Juan Carlos Correa. "Comparación Entre Árboles De Regresión CART Y Regresión Lineal." *Comunicaciones en Estadística* 6, no. 2 (2013): 175-195.

[24] Cadenas, José M, M Carmen Garrido, and R A Díaz-Valladares. "Soft Computing En Ensambles Basados En Boosting, Bagging Y Random Forest." *Rev. Iberoam. de Sistemas, Cibern. e Informática* 7, no. 1 (2010): 25-32.

[25] Proyecto Boss Multitel: http://www.multitel.be/image/research-development/research-projects/boss.php

[26] J. A. Parker, R. V. Kenyon and D. E. Troxel, "Comparison of Interpolating Methods for Image Resampling," in *IEEE Transactions on Medical Imaging*, vol. 2, no. 1, pp. 31-39, March 1983.

[27] Harel, Jonathan & Koch, Christof & Perona, Pietro. (2006). Graph-Based Visual Saliency. Adv. Neural Inform. Process. Syst. 19. 545-552.

[28] Farnebäck, Gunnar. "Two-frame Motion Estimation Based on Polynomial Expansion." *Image analysis* (2003): 363-370.

[29] AWANO, Naoyuki. "Acceleration of Calculation for Field of Visual Perception on Digital Image and Verification of the Application to Halftone Image." *Transactions of Japan Society of Kansei Engineering* 16, no. 2 (2017): 209-218.

[30] Markus Bindemann, Scene and screen center bias early eye movements in scene viewing, In Vision Research, Volume 50, Issue 23, 2010, Pages 2577-2587

[31] Rapid object detection using a boosted cascade of simple features
P Viola, M Jones - Computer Vision and Pattern Recognition, 2001

[32] Jing Zhong and S. Sclaroff, "Segmenting foreground objects from a dynamic textured background via a robust Kalman filter," *Proceedings Ninth IEEE International Conference on Computer Vision*, Nice, France, 2003, pp. 44-50 vol.1

[33] Bylinskii, Zoya, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. "What Do Different Evaluation Metrics Tell Us About Saliency Models?" *arXiv preprint arXiv:1604.03605* (2016).

[34] Leboran, Victor, Anton Garcia-Diaz, Xosé R Fdez-Vidal, and Xosé M Pardo. "Dynamic Whitening Saliency." *IEEE transactions on pattern analysis and machine intelligence* 39, no. 5 (2017): 893-907.

[35] Breiman, L., J. Friedman, R. Olshen y C. piedra. *Clasificación y árboles de regresión*. Boca Raton, FL: CRC Press, 1984.

[36] Loh, W.Y. "Árboles de regresión con selección de variables objetiva y detección de interacción."*Statistica Sinica*, Vol. 12, 2002, pp. 361-386.